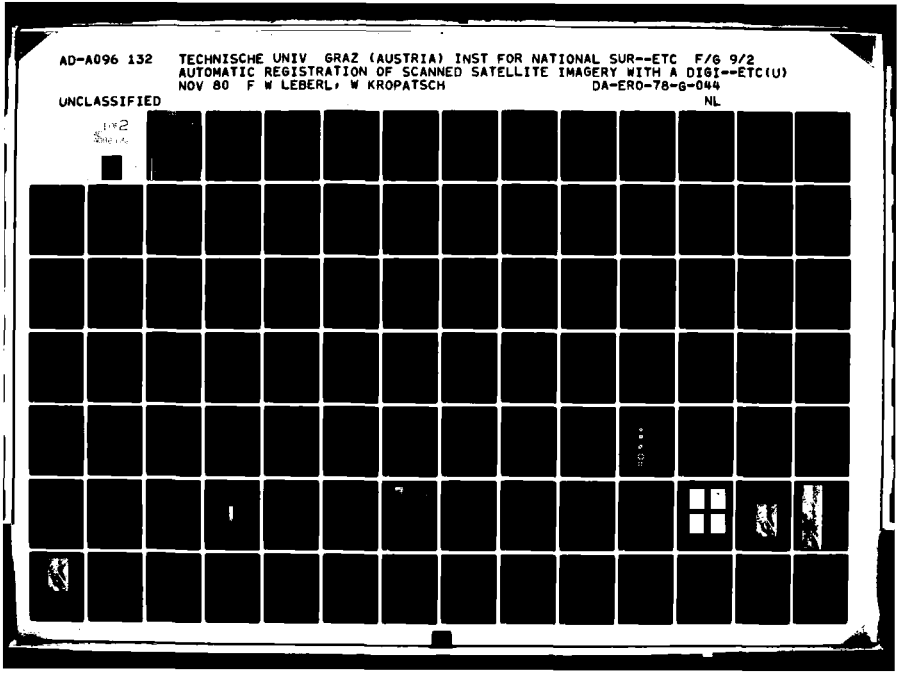
AD-A096 132 TECHNISCHE UNIV GRAZ (AUSTRIA) INST FOR NATIONAL SUR--ETC F/G 9/2
AUTOMATIC REGISTRATION OF SCANNED SATELLITE IMAGERY WITH A DIGI--ETC(U)
NOV 80 F W LEBERL, W KROPATSCH DA-ERO-78-G-044

UNCLASSIFIED NL



LEVEL II

AUTOMATED REGISTRATION OF SCANNED SATELLITE IMAGERY WITH A DIGITAL MAP DATA BASE

(6)

Final Technical Report

by

Franz W. LEBERL and Walter KROPATSCH

September 1980

EUROPEAN RESEARCH OFFICE
United States Army
London , England

GRANT NUMBER DAERO - 78-G-044
Grantee

ıstitute for National Surveying and Photogrammetry
Technical University Graz
A-8010 Graz , Austria

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A096 432
	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) Automatic Registration of Scanned Satellite Imagery with a Digital Map Data Base.	5. TYPE OF REPORT & PERIOD COVERED (9) Final Technical Report. Feb 78 — Nov 80,
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Franz W. Leberl ~~and~~ Walter Kropatsch	8. CONTRACT OR GRANT NUMBER(s) (15) DAERO-78-G-044
9. PERFORMING ORGANIZATION NAME AND ADDRESS Institute for National Surveying and Photogrammetry, Technical Univ. Graz A-8010 Graz, Austria	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 6.11.02A (17) 1T161102BH57-01 (16)
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Research, Development & Standardization Group-UK, Box 65, FPO NY 09510	12. REPORT DATE (11) Nov 80
	13. NUMBER OF PAGES (12) 259
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution unlimited - Approved for Public Release

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Photogrammetry, Automatic registration, satellite imagery, digital imagery, digital maps, map-guided interpretation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Analysis of digital remote sensing images can, and should, employ all the information that is available about the area of interest. Maps often do exist, so that the question arises how map information can be automatically used to direct image analysis. With an appropriate organisation of digital map data banks it becomes possible to merge image and map. Attention is concentrated on areal features.

NL 412 ...

The algorithms and data structures used for the task are described and experiences using specific examples of Landsat images in Southern Germany and Austria to automatically recognize features for subsequent rectification are reported.

TABLE OF CONTENTS page

page

page

ABSTRACT

Analysis of digital remote sensing images can, and should, employ all the information that is available about the area of interest. Maps often do exist, so that the question arises how map information can be automatically used do direct image analysis. With an appropriate organisation of digital map data banks it becomes possible to merge image and map. So far we have concentrated on areal features. We describe in the report the algorithms, data structures used for the task and we report on our experiences using specific examples of Landsat images in Southern Germany and Austria to automatically recognize features for subsequent rectification.

RESUME

L'analyse des images numériques de télédétection doit utiliser toute l'information qui existe d'une region. Les cartes géographiques normalement existent; il se pose la question comment utiliser ces cartes, d'une manière automatique, pour diriger l'analyse d'une image. Une banque de données numériques doit etre arrangée d'une manière specifique pour relier image et carte. Jusqu'à présent tous nos efforts se concentraient aux objets aréaux. Dans le rapport nous decrivons algorithmes, la structure des données, et nous discutons les expériences obtenues en reconnaissant automatiquement des objets dans des scenes Landsat, située au sud de l'Allemagne et en Autriche, et son application au redressement des images.

ZUSAMMENFASSONG

Die Auswertung digitaler Fernerkundungsbilder soll auf der gesamten Information beruhen, welche über das Interessensgebiet zur Verfügung steht. Meist bestehen Karten, sodaß sich die Frage erhebt, wie die Kartendaten für eine Steuerung der Bildanalyse in automatischer Weise verwendet werden können. Mit einer geeigneten Organisationsform für die digitalisierten Karteninhalte in einer Kartendatenbank wird es möglich, Karte und Bild zu verschmelzen. Bisher haben wir uns auf flächenhafte Merkmale in Bild und Karte konzentriert, um entsprechend Bild und Kartenmerkmale automatisch aufzufinden. Wir beschreiben in der vorliegenden Arbeit unsere Methoden, Datenstrukturen und Erfahrungen mit Beispielen von Landsat-Aufnahmen in Süd-Deutschland und Österreich in der automatischen Erkennung von Merkmalen zur nachfolgenden Entzerrung.

1. INTRODUCTION

The analysis of remote sensing imagery should rely on available data from the investigated area.Generally this will be so called *ground truth* in the form of observations made in the field or of imagery of higher diagnostic power, e.g. of a larger scale. In many instances, however, remote sensing imagery is being analyzed of areas where excellent maps already exist. A number of investigators have thus come up with procedures to merge digital imagery with the existing digitized map so that the interpreter/user can take full advantage of both data sources. So far, these procedures were essentially manual, with the purpose of merely presenting a combined data set (BRYANT and ZOBRIST, 1977).

The question arises naturally, how a procedure must operate that merges digital images and maps automatically and how the data present in the map can help to automatically analyze a digital image.

We have taken it upon us to study these questions. Obviously they address a very complex area that falls into the category of *automatic photo-interpretation*. We are well aware of the fairly pessimistic opinions sometimes expressed on the prospects of automation of photo-interpretation. Nonetheless do we believe that it makes sense to support human analysis of digital images by automation and to incorporate an available map into this process. There are certain monitoring tasks such as for example in the detection of changes that could be nearly fully automated in this way; for other remote sensing applications this may not be the case; the benefits of automation for those applications may consist of greater efficiency of the interaction between man and machine.

Map-guidance in the image analysis is central in our effort. The term *map-guided automatic photo-interpreation* has previously been used by BARROW et al. (1977). Essentially we see in our work three areas of emphasis:

- the organisation of the digital map data;
- the analysis of areal features;
- the analysis of linear features.

The tools required encompass a range of manipulating digital graphics data, of image processing and of automatic pattern recognition. Our general approach to automatically merge image and map was described in a previous paper (KROPATSCH and LEBERL, 1978 a). We will review this concept in the next section. This will be followed by a description of the manipulation of map data (chapters 3 and 4), of a discussion of image pre-processing (chapter 5), and the establishment of relationships between maps and image (chapter 6).

In our study we employ the techniques of automated feature recognition to the geometric image rectification. A description of these fairly common procedures follows. Finally, we go into our results as obtained with Landsat digital satellite images.

Previous work in this area has been performed by STIES et al. (1977); they described a comprehensive image interpretation system that includes a map data bank suitable for map-guided image analysis. BARROW et al. (1977) and TENENBAUM et al. (1978) have mainly concentrated on the analysis of linear features in digital images. With satellite images at small scales, however, it is promising to study mainly areal features. After all, satellite remote sensing draws its essential motivation from the analysis of natural surface features: it is for this reason that scales can remain comparatively small, while the study of manmade linear objects would require larger scales and thus airborne remote sensing.

However, some results have been obtained also with linear features from satellite scanner data.

It should be emphasized, that the techniques may have a broad spectrum of uses, not only for rectification but also for other tasks. As an example one may point to the work of FLONZAT et al. (1979) who based multi-spectral classification on training areas which are taken from a map and transformed into the image.

If we go back somewhat we find that an early automatic image recognition application was the measurement of pricked points (KREILING, 1976) and of reseau points (ROOS, 1975).

The associative, complex mental interpretation of an image by the human expert will hardly ever be matched by the computer. However, specific tasks may be ideally suited for automation. In the current context we are certainly not tackling the most trivial application but one which in its complexity helps one to understand the complexity of the problem in general. We may take the statement of a prominent pattern recognizer for whatever it is worth, but ALEKSANDER (1978) claimed, that "most problems of automated pattern recognition have solutions; difficulties only exist in the speed and general applicability of these solutions". Our current study is aimed at the understanding of solutions to the automated recognition of map features in images. Our first application is to use these features as control points for a geometric rectification.

2. GENERAL CONCEPT OF ARSIM

The task of ARSIM (ARSIM is an acronym for Automatic Registration of Satellite Images and Maps) consits of registering a given satellite image with the corresponding map. Because of local distortions of the image the registration cannot be done with a simple global transformation defined a priori. A common way to register images with each other is the use of control points. This method is also suitable for the present task. It has first to produce enough control points so that one can rectify the image geometrically. Figure 2.1 presents a general program-flow and shows that

```
(1)    program ARSIM
(2)    begin
(3)        REGISTER (map, image, control points);
(4)        if sufficient control points then
(5)            RECTIFY (image, control points, new image)
(6)                                        else
(7)            print ('map and image cannot be registered');
(8)    end.
```

Figure 2.1: Main program for project ARSIM.

the problem splits into two mainly independent procedures REGISTER, which registers map and image delivering control points, and RECTIFY, which rectifies the geometry of the image with the aid of the control points to result in a new image. The new image has then the geometry of the map. The method of operation is described in detail in chapter 7.

Before the processing of the lower level procedures is described in more detail an overview of the registration process is discussed by means of a flow-diagram (figure 2.2).

In this flowchart processing steps (1) to (9) correspond to the procedure REGISTER, while step (10) correspond to procedure RECTIFY. The following chapters 3 through 7 describe the functions of the lower level procedures in detail mostly using a general purpose Problem Description Language (PDL).

2.1 THE REGISTRATION PROCESS

Following NACK (1975) we define in the digital image a square (rectangular) Area of Interest (AI) that is being processed at any given time.

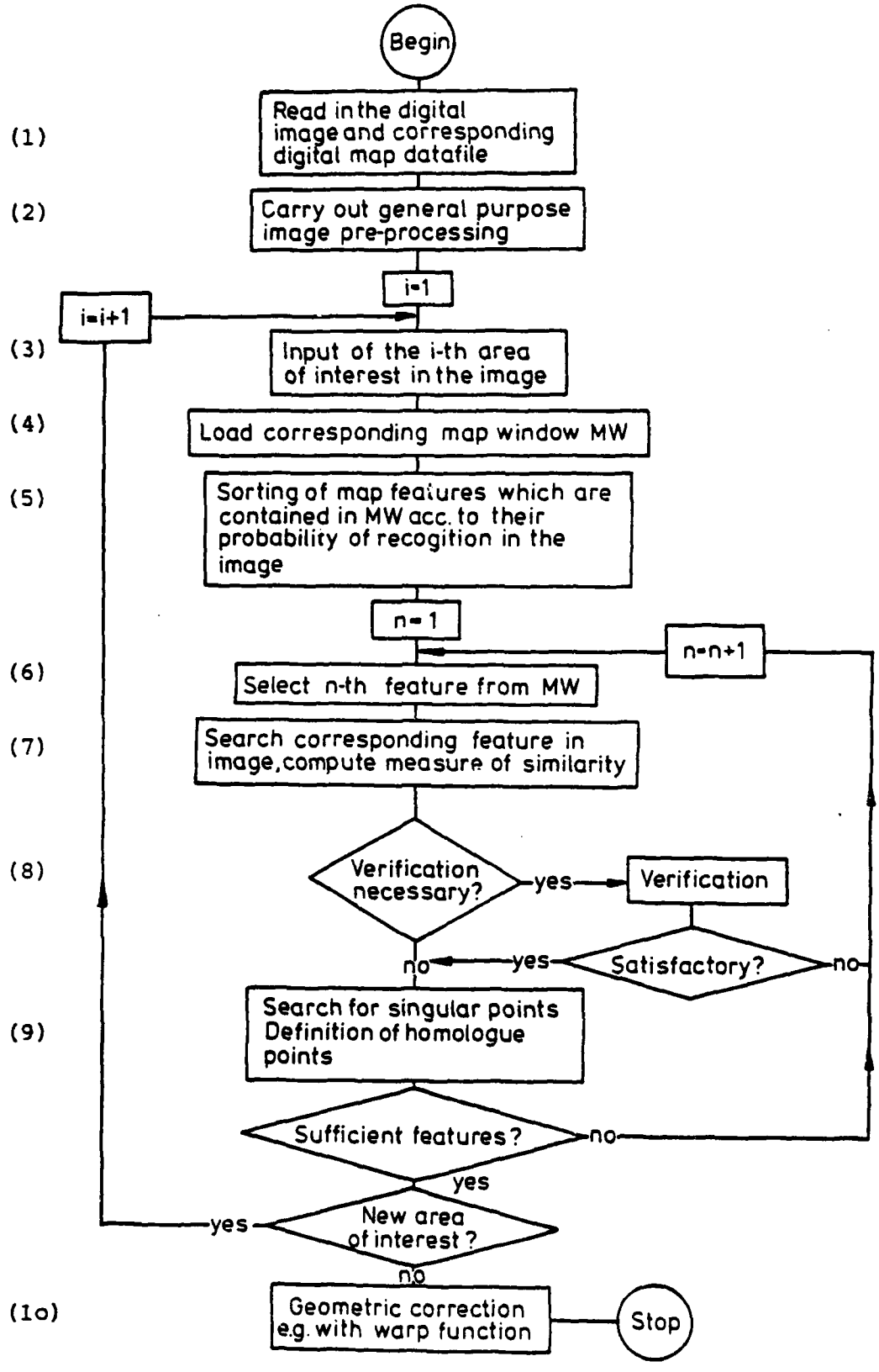


Figure 2.2: Flow-diagram of the operation of ARSIM.

For processing-time considerations the digital image to be registered with the map devides into a number of AIs. These are easier to handle than the total image. They restrict the search and speed up the procedure. This logical processing unit subdivides further the operations of procedure REGISTER into

(A) operations with AIs (steps (3) until (6) in figure 2.2), and

(B) operations within AIs (steps (7) until (9) in fig. 2.2).

Operations (A) form procedure REGISTER (figure 2.3), the operations from (B) are collected in a subroutine called OVERLAY and described in chapter 2.2.

```
(1) procedure REGISTER (map, image, controlpoints);
(2) begin
(3)   controlpoints:= empty;
(4)   repeat
(5)   begin
(6)      SELECT (AI); $ input an Area of Interest $;
(7)      TRANSFORMWINDOW (AI, MW);
(8)      MAPLOAD (map, MW);
(9)      PREPROCESS (image, AI);
(10)     OVERLAY (image, AI, map, MW, controlpoints AI);
(11)     controlpoints: = controlpoints+controlpoints AI ;
(12)  end until no input left;
(13) end.
```

Figure 2.3: Operations with Areas of Interest (AI).

In step 6 (figure 2.3) we start from a digital map base that is available in a format compatible with the image processing system DIBAG, which contains the satellite image to be registered. SELECT (AI) reads an AI, which can be chosen manually by the following criteria:

- it has a predefined window width,
- in that window there is a certain amount of patterns, which can be optically distinguished in the image,
- for most of these patters in the image exist corresponding objects in the map.

Each pixel of a digital satellite image can be located in a map projection with a certain positional accuracy. For LANDSAT, maximum errors have been demonstrated to amount to not more than 5 km (COLVOCORESSES et al., 1973). Data used to achieve this accuracy are the satellite position and attitude values delivered with each image. More recently, orbit parameters in LANDSAT permit an even better accuracy to be achieved without ground control data, namely about 10 pixel diameters or 800 m. Using the map locations of the corner points of AI we define the corresponding map window (MW) (procedure TRANSFORMWINDOW

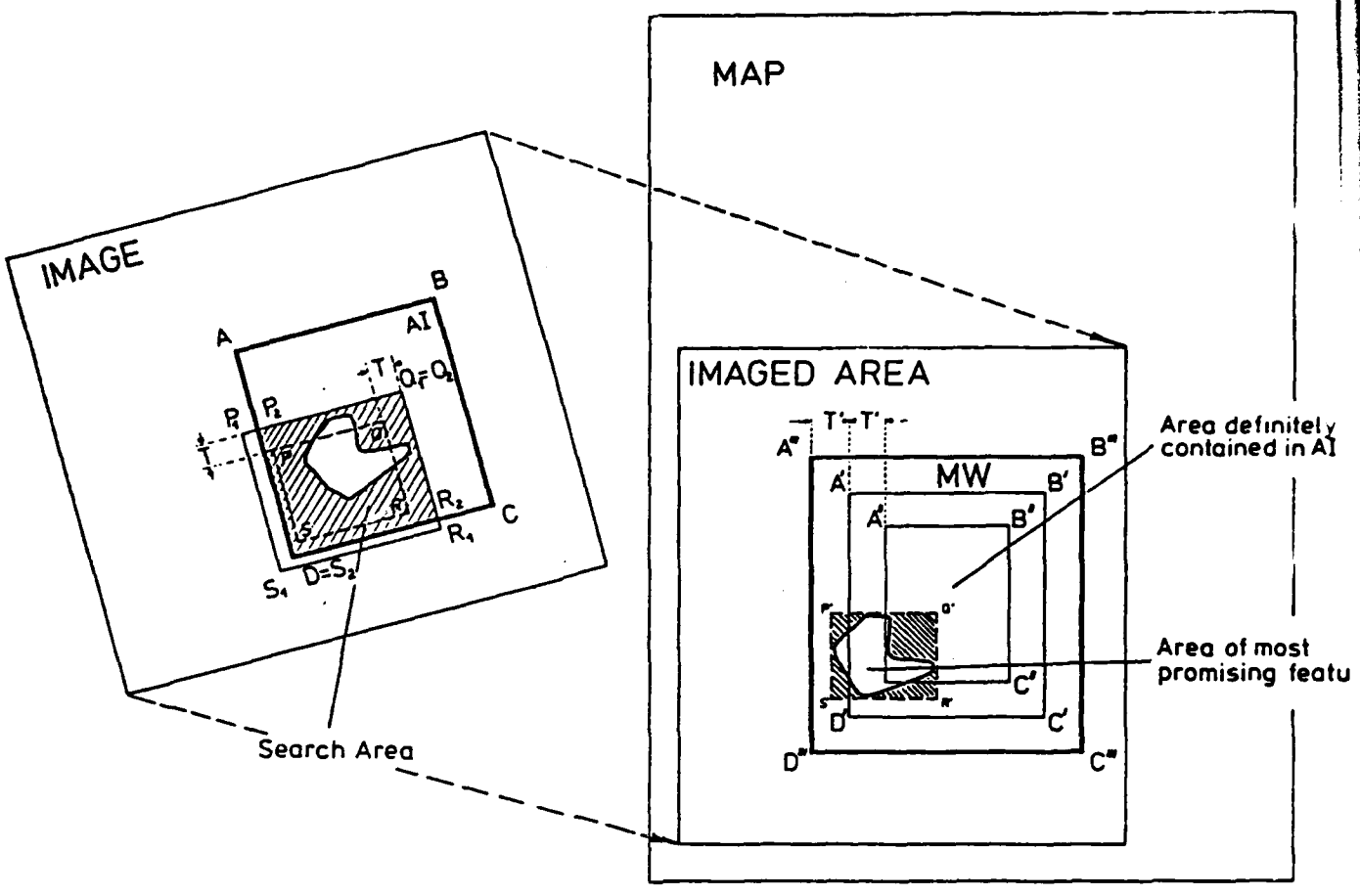


Figure 2.4: Search areas for ARSIM in the image and map. For explanation of the sequential searches compare table 2.1.Distance T (and T') serve to account for the geometric errors of the satellite image.

TRANSFORMWINDOW	(1) A B C D $\in$ Image $\longrightarrow$ A' B' C' D'	$\in$ Map
	(2) A'B'C'D' + T' $\longrightarrow$ A" B" C" D"	= Map window MW
step 5 in Figure 2.2	(3) A'B'C'D' - T' $\longrightarrow$ A"'B'"C'"D'"	$\in$ Map
	(4) Feature sorting $\longrightarrow$ P' Q' R' S'	$\subset$ A'" B'"C"'D"'
step 6 in Figure 2.2	(5) P'Q'R'S' $\in$ Map $\longrightarrow$ P Q R S	$\in$ Image
	(6) P Q R S + T $\longrightarrow$ P_1 Q_1 R_1 S_1	$\in$ Image
	(7) $P_1Q_1R_1S_1 \cap$ ABCD $\longrightarrow$ P_2 Q_2 R_2 S_2	$\in$ Image: Search Area

Table 2.1:ARSIM - search areas as described in figure 2.4.

in step (7) of figure 2.3). In order to contain all of AI, the window has to be larger than AI due to the limited positional accuracy (compare figure 2.4).

Then MAPLOAD loads a portion of the map data bank which comprises only objects completely in MW (step 8 of figure 2.3). The structure of the data bank is described in chapter 3.

Depending upon the types of objects contained in that portion of the map we are now able to call routine PREPROCESS (step 9 of figure 2.3) for the AI in the image. For each object-type there exists a special method of pre-processing of the image data (for more detail see chapter 5),so that when we search for some object we do it on an image that is pre-processed for this object-type.

2.2 REGISTRATION IN AN AREA OF INTEREST.

After all the preparations and restrictions we are now able to turn to the main task: identification of objects from the map in the image. This is done in procedure OVERLAY and consists of 5 major operations as shown in figure 2.5:

- the selection of an object from MW (OBJECTSELECTION);
- the extraction of the object from the data bank and its transformation in a format which allows a comparison with the image (MAPTRANSFORM);
- the recognition of the object in the image (RECOGNIZE);
- the VERIFICATION that the pattern found in the image corresponds to the original object and
- the search for controlpoints (CONTROLP).

```
(1)  procedure OVERLAY (image, AI, map, MW, controlpoints);
(2)  begin
(3)     controlpoints: = empty;
(4)     OBJECTSELECTION (map, MW, list);
(5)     while list not empty do
(6)     begin
(7)        object: = first (list);
(8)        list: = list - first (list);
(9)        MAPTRANSFORM (map, object, binarymatrix);
(10)       RECOGNIZE (image, AI, binarymatrix, pattern);
(11)       if VERIFICATION (object, image, pattern)>accepttresh then
(12)          CONTROLP (object, pattern, controlpoints)
(13)    end;
(14) end.
```

Figure 2.5: Overlay of image AI and MW.

Features within MW are sorted according to their probability of being identifiable in the satellite image. This probability can be (automatically) adjusted as the result of previous

experiences. Consequently, the procedure has a learning capability. The procedure OBJECTSELECTION (figure 2.6) provides a sorted list of these features. The most probable feature within MW is selected and a search begins in the image-AI to find the corresponding detail. MAPTRANSFORM (see figure 4.2) extracts a boundary polygon of this "object" from the map data bank, it linearly transforms the polygon into the image coordinates, using the same function as in TRANSFORMWINDOW and builds a binary matrix which determines if a pixel of the AI is either inside the polygon or outside. The binary matrix of the object serves as input to procedure RECOGNIZE (figure 2.7). It breaks down the recognition process into 4 different methods, which are described in chapter 6. The result of the search is another binary matrix called "pattern".

```
(1) procedure OBJECTSELECTION (map, MW, list);
(2) begin
(3)     list: = empty;
(4)     for all objects ∈ MW do
(5)     begin
(6)         recogn: = probability (object);
(7)         if recogn > recogthresh then list:= list +(object,recogn);
(8)     end;
(9)     SORT (list)   ;
(10) end.
```

Figure 2.6: Selection and Sort of objects in MW.

Next the identity of the map and image feature has to be verified by a comparison of the "object" and the "pattern" (VERIFICATION).A verification measure is compared to a certain "accepttresh" to include the worst case: a reject.

A last step (12 in figure 2.5) is the search for singularities of the feature so that unique homologue pairs of coordinates can be identified in both the image and the map (CONTROLP). As this is an essential concept for the rectification process, chapter 7 contains a description of it.

```
(1) procedure RECOGNIZE (image, AI, binarymatrix, pattern)
(2) begin
(3)     case search method (object) of
(4)     begin
(5)         1: SHIFT (image , binarymatrix, pattern)
(6)         2: THRESH (image,  binarymatrix, pattern)
(7)         3: ADAPT (image,  binarymatrix, pattern, D)
(8)         4: LINDET (image, binarymatrix, start (object),
                              goal   (object), pattern);
(9)     end
(10) end.
```

Figure 2.7: Recognizing an object by different methods.

3. OPERATIONS WITH MAP DATA

3.1 GENERAL

Processing of digital map data is basically different from that of images; we are not dealing here with raster type data, but with vectors. The area of work is often times denoted by the expression "computer graphics" and is seen to be the fastest growing portion of the computer industry.

In the current context we are dealing with computer graphics in a specific way: we are processing vector data and raster information and need conversion of one into the other. This is not a trivial task, even if it may seem so at superficial inspection.

Graphical data banks consist of data and a computer program system for manipulation of the data. A vast body of literature exists in this area, also from the particular point of view of cartography (e.g. FRANK, 1979; WEBER, 1978). Depending on specific requirements and uses data banks may be simple or very complex. Data structures define complexity and in turn result from the type of relationships that must be defined among elements of the data bank.

Data structures can roughly be grouped into two groups:

(a) one without explicit definition of relationship among data elements (e.g. object oriented sequential or raster data structure);

(b) one with explicit definition of relationship (topology, neighbourhood, positional reference).

Most of currently implemented cartographic data banks belong to category (a) without a thorough definition of relationship among data elements.

In our current application we face the task of defining a map data bank of category (b). It must thus be possible to establish very quickly interconnections, neighbourhoods, relationship among objects of a map, much like the process of a visual inspection of map contents by a human interpreter. A topological data structure will thus be required. It is straight-forward to define a set of functions to be satisfied by the data bank:

- selection and extraction of single elements (objects) of the data bank, using names, properties, positions;
- establishment of connections between individual objects in geometrical and thematic set operations;
- computations with elements;
- generation of lists and printouts;
- graphical output in vector or raster format.

This enumeration may not be complete but represents a set of important features to be had by the data bank in its application to work with maps and images.

We will describe in this chapter the basic data structures, algorithms and uses of the data bank. Detail is appropriate to instruct a prospective user of the data bank. However, it needs to be stressed that the data bank is experimental in nature, and the current implementation is motivated solely be the need to experiment with combined sets of image and map data. Numerous improvements will have to be made in the areas of user friendly data acquisition and editing.

3.2 DATA STRUCTURES

3.2.1 General

We deal with data that are available in the form of strings of x,y-coordinates representing lines with a specially identified beginning point. It is not relevant here how these points were digitized - be it manual digitizing, line following or raster scanning with subsequent vectorisation.

There exists a multitude of data structures for various types of graphical data banks. Methods to store those features of a map in digital form that are relecant in our current problem could be:

(i) Each feature is represented by its contour polygon (object-oriented, sequential data structure);

(iii) Each feature is defined by a sequence of line segments. Each segment separates exactly 2 features and consists itself of a polygonal sequence of points. The map represents a "planar graph".

(iii) Raster presentation;

(iv) Geo-coded or positionally defined data, with the special case of the quad tree[1] (WEBER, 1978; DYER et al., 1979);

(v) Description using the Freeman-Chain Code[2] (FREEMAN, 1979).

Since we require a connection between map data and digital images we need to stress that these images exist in the form of matrices of gray values. This corresponds to the concept of a raster structure for graphical data as used by WEBER (1978), BRÜGGEMANN (1978), HARRIS (1979), HARRIS and PRESTON (1979), GOODCHILD (1979) and others. It would thus be logical to consider structure (iii) for a data bank.

---

1) "Quad Tree" is the name of a tree structure of 4th order. Its root is a square of the base plane containing the entire feature. There exist four "sons" of a "node" by subdividing the square in 4 equal segments. In each "leave" of the tree (i.e. in a node without successor) one finds whether its square belongs to the basic square in the base plane or not.

2) This is based on a raster representation. It describes the contour of a feature or object using a sequence of small vectors (standardized and coded) which connect adjacent raster points.

However, with the map features in a raster format one encounters immediately problems due to storage, redundancy, differences of raster mesh size and orientation and complexity of required transformation routines.

If we evaluate the five classes of data structures we find that an object-oriented, sequential method (i) does not compare well with method (ii) because of redundancy: all lines must be digitized and stored twice.

Reference to position in method (iv) appeals due to its efficiency when addressing points or features in a two-dimensional space. Judging method (v) shows that this is suitable particularly in cases where data were digitized in a raster format by a scanner. Current assumptions are, however, that data exist in the form of vectors. But method (v), the Freeman-Chain Code, still has an advantage of reduced storage requirements. This is set off by increased difficulties for geometric transformations.

These various data structures are not mutually exclusive. A feature-oriented, topologically meaningful structure acc. to (ii) can exist along with a positional one such as (iv). This then would permit one to get an answer to two questions like:

- Where is a specific feature and what are its neighbours?
- What feature exists at a specific position?

3.2.2 A structure to merge map and image

Prior to a decision on the data structure one needs to identify the principles of procedures to merge map and image. We do see two main alternatives:

(a) A synthetic image is generated from the graphical data; actual and synthetic images are then conventionally cross-correlated.

(b) Map data is structured into elementary information, namely features or objets. Of these one selects specific ones for a meaningful connection with a given image.

The first route must fail first of all when maps contain features that do not appear in an image: political boundaries, for example, or in the reserve case, clouds; furtheron if map features are generalized as with roads. We must realize that maps have primarely been produced to portray the earth's surface for interpretation by a human. We therefore turn our attention preferably to the second method (b) above.

One not only has available for a merging process the shape and position of an object, but also its topological environment. A rock area, for example, is more difficult to identify in an image if it is surrounded by glaciers as compared to forests.

This leads then to the use of a topological data structure (WEBER, 1978) as the highest level of the data bank, according to method (ii) in section 3.2.1. This structure will be discussed in detail later. The highest level of the structure presents the position of features with respect to one another. There still remains the task to store thematic information and to exactly position the feature (geometric information). Since the topological relationships have been defined already in the highest level one can make use of the simplest data structure for the lower levels; this is the sequential one. It is here that features are described geometrically and thematically.

A data bank is thus organized in three levels of information (Fig. 3.1):

- topological relationship (structure);
- sequential lists of objects, i.e. of features (regions) and of lines (region list, line list);
- polygons stored as a sequential coordinate list (coordinate file).

The next three sections will present the data bank, followed by two data structures useful for the current problem: the coordinate list of an object's contour, and the binary matrix. The former serves as an intermediate storage and for transformation of coordinate systems; the latter is needed for geometric set operations.

This structure also permits search processes based on a positional reference, however, in an inefficient manner. For improved economy of this type of search processes one would have to define the positional reference set over the topological level.

3.2.3 The topological structure, the graph.

In a formal manner one can interpret a map to be a planar graph[1]:

$$G = (R,L)$$

R ... set of regions (features, nodes);
L ... set of lines (edges), separating 2 regions.

---

[1]) Definition: a (finite) graph G = (X,U) is composed of a pair, consisting of the finite set X, the "nodes", and a set U c X x X, the "edges". A graph is "planar" if it is possible to represent it in a plane in such a way that all nodes are separable and that all edges are simple curves, without intersections or double points, and that two edges only meet at the end points (SAKAROVITCH, 1975).

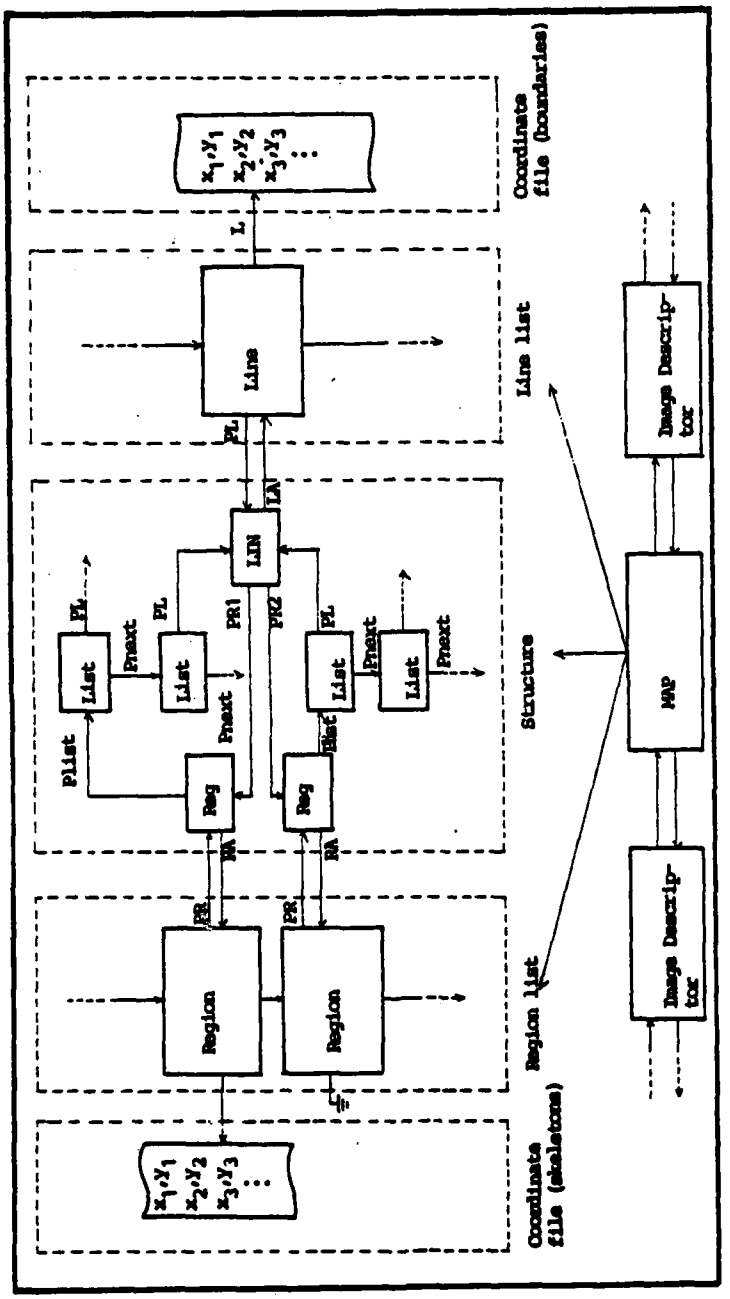


Figure 3.1: Organisation of the digital cartographic data bank. "Structure" denotes the graph of the map features.

The graph should be stored digitally on a minimum of storage since all inputs/outputs of data are based on this structure. This can be achieved by a reduction and compression of the information kept in the smallest storage units, so called items. Information is reduced to pointers describing topological relationships. There are 3 types of items (REG, LIN, LIST):

REG: = (① pointer RA to a region list; ② pointer PLIST to a LIST-item).

LIN: = (① code 262143[1] ; ② pointer LA to the line list; ③ pointer PR1; ④ pointer PR2).

LIST: = (① pointer PL to a LIN-item; ② pointer PNEXT to the next LIST-item).

Pointers PR1 and PR2 point to two REG-items which correspond to the regions separated by a line.

On a 36-bit-computer each REG-item exists of 1 computer word (2 halfwords), each LIN-item of 2 words (4 halfwords) and each LIST-item of 1 word (2 halfwords).

The mode of data acquisition and properties of features lead to definition of two types of regions:

- areas and
- skeletons.

Skeletons are areas represented by lines such as rivers or roads. Two reasons exist for this separate definition of skeletons:

(a) they permit digitizing of such linearly extended areas as if they were lines; there is no need to digitize all contours of such an area - this would be a redundant effort;

(b) the width of skeletons can be selected at a later stage - a feature of importance when various scales are needed in visualizing the map contents.

Skeletons differ from areas in the graph: the pointer LA of a border line is replaced by a specific code (the number 262142 in our case). This can be done because in the case of a skeleton there are no specifically digitized border lines.

---

[1]) Redundant code to fill up 2 computer words. Denotes a LIN-item.

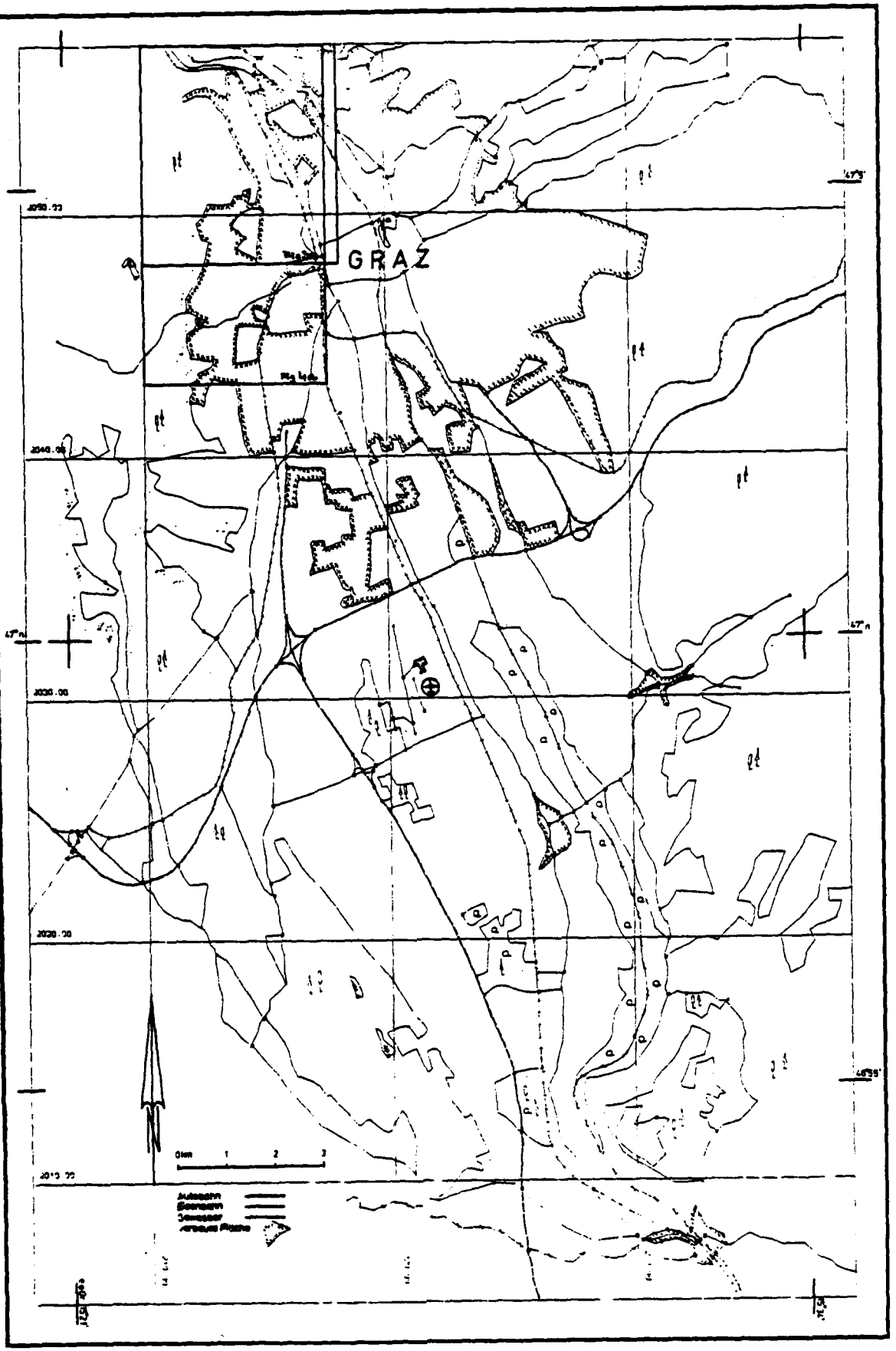


Figure 3.2: Digital map of Graz

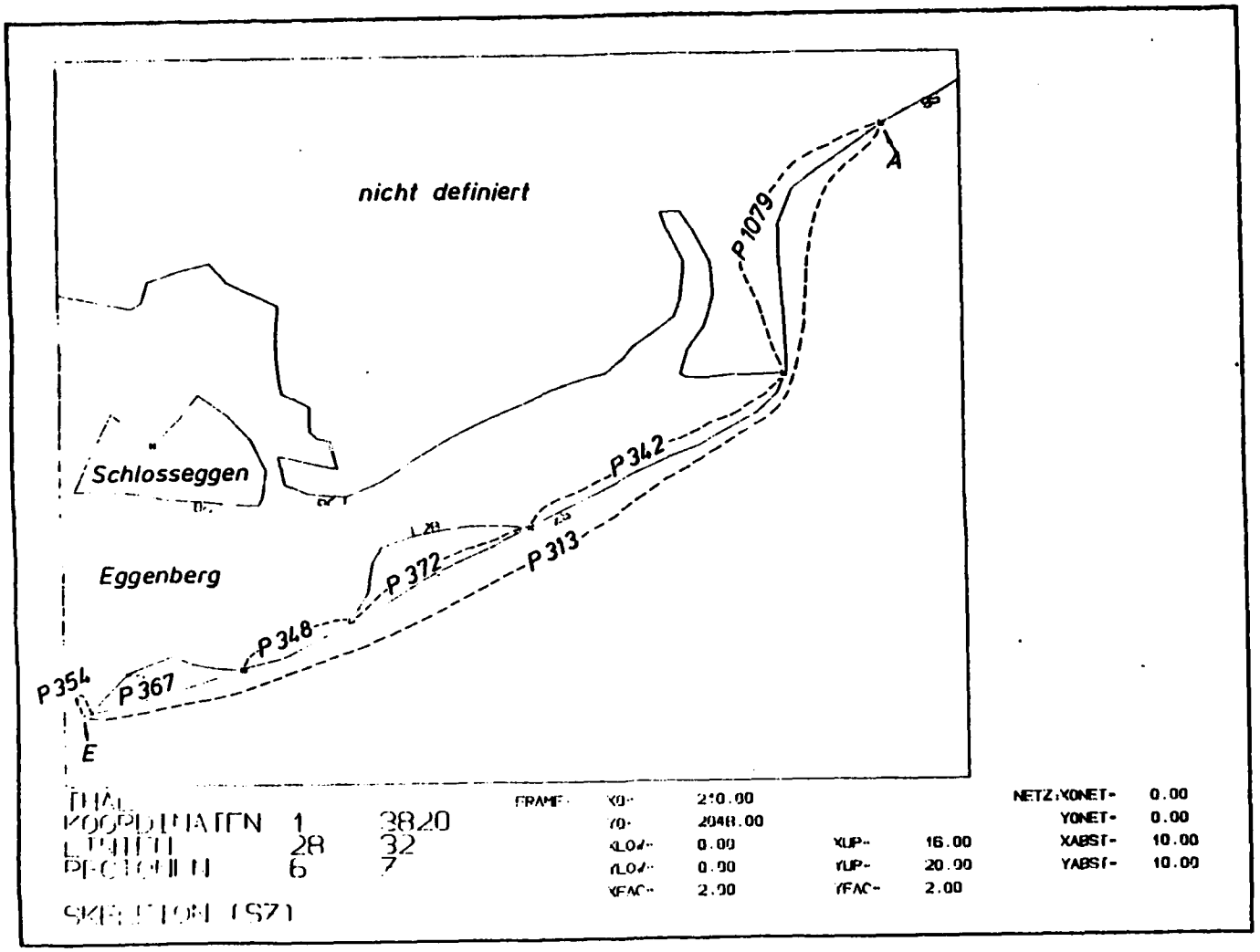
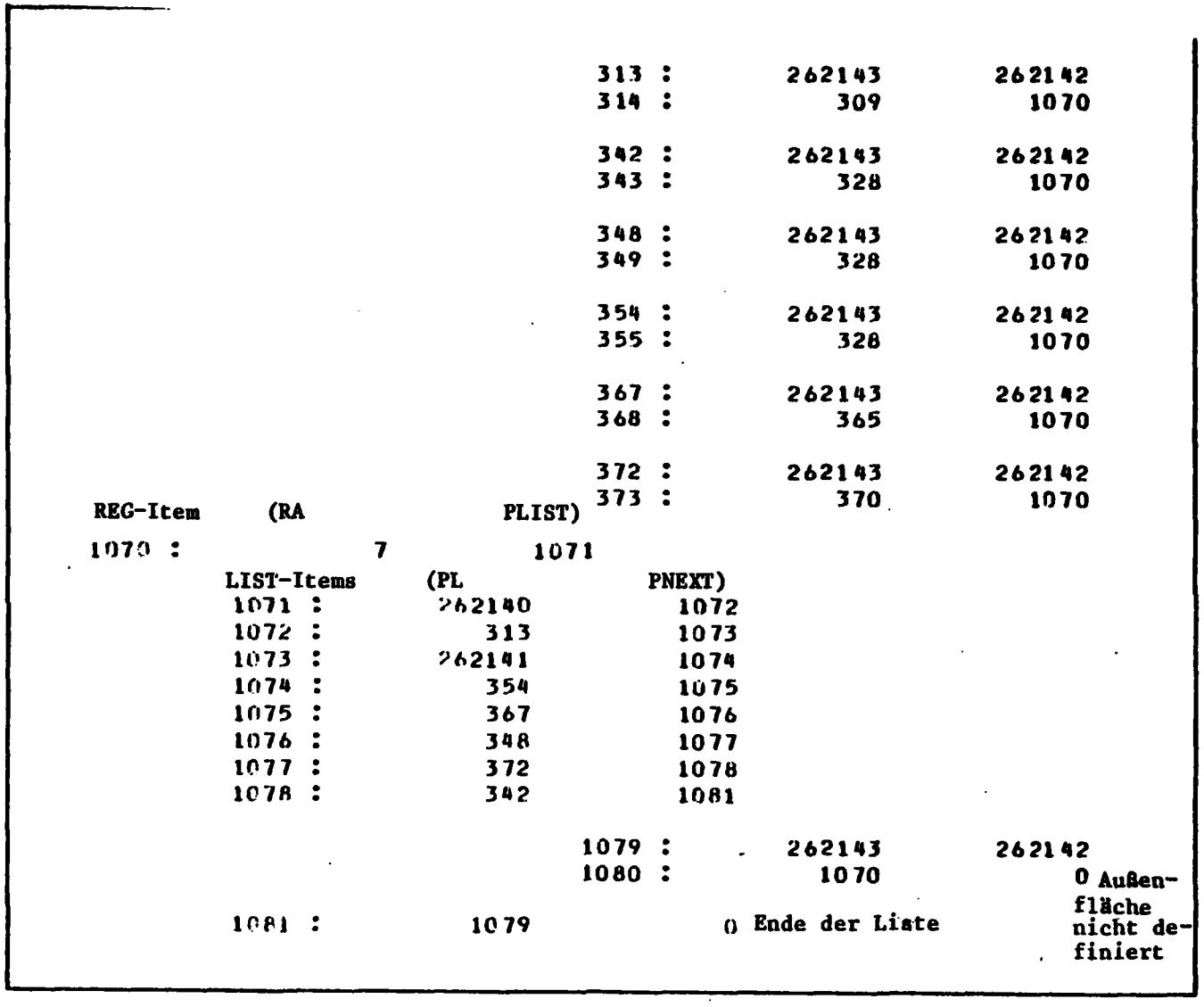


Figure 3.3(a): Section of map data bank of Graz around skeleton S7.



	313 :	262143	262142
	314 :	309	1070
	342 :	262143	262142
	343 :	328	1070
	348 :	262143	262142
	349 :	328	1070
	354 :	262143	262142
	355 :	328	1070
	367 :	262143	262142
	368 :	365	1070
	372 :	262143	262142
	373 :	370	1070

REG-Item	(RA	PLIST)
1070 :	7	1071

LIST-Items	(PL	PNEXT)
1071 :	262140	1072
1072 :	313	1073
1073 :	262141	1074
1074 :	354	1075
1075 :	367	1076
1076 :	348	1077
1077 :	372	1078
1078 :	342	1081

1079 :	262143	262142	
1080 :	1070		0 Außenfläche nicht definiert
1081 :	1079	0 Ende der Liste	

Figure 3.3(b): Graph pertaining to skeleton S7 of figure 3.3(a).

However, the graph needs a LIN-item: this is then a pseudo-line which was not digitized. A skeleton is thus an area with pseudo-lines as contour. Beginning and end-points of a skeleton are two specific LIST-items where PL-pointers are replaced by a code for the beginning (the number 262140 in our case) and end (the number 26241). Figure 3.2 illustrates a portion of a data base in the area of Graz, digitized off maps at scale 1:50 000. Elements of this data base will be used to illustrate the topological data structure.

3.2.4 Illustration of the graph using a skeleton.

Taking a specific skeleton from fig. 3.2 we describe its pertinent structure in figures 3.3 (a)and 3.3(b).

Fig. 3.3(a) presents a skeleton extending between points A and E, and with the name S7. As a region or area it is delimited by a series of pseudo-lines P313, P354, P367, ..., P1079. A printout of the graph is shown in fig. 3.3(b), with only the data rekeating to skeleton S7. Rows or lines in the computer printout are shifted left-right depending on whether it concerns a LIN-, LIST- or REG-item. Each line of the printout contains a line number, increasing regularly from 1 to n, and the content of a 36-bit-computer word with two integers. Each LIN-item consists of 2 rows, each REG- and LIST-item of one row.

To explain the printout in more detail, let us take the first item belonging to skeleton S7. It is in rows 313 and 314 of the graph, with the following format:

Row number 313:	262143	LA
Row number 314:	PR1	PR2

This is a LIN-item as is obvious from code 262143. The value of LA is 262142; we deal this with a pseudo line separating two regions. The first region has its REG-item in row 309 as decumented by the value 309 of pointer PR1, the second region in row 1070 (see pointer PR2). Figure 3.3(b) presents all LIN-items relating to the second region with REG-item in row 1070. Other rows of the printout were suppressed. REG-item 1070 has an RA-pointer 7 and therefore represents skeleton S7 (compare the region file, section 3.2.6).

Then LIN-items in rows342, 348, 354 separate the region in row 1070 from a region in row 328. In the region file that region has the name EGGENBERG.

LIST-items serve to accelerate the search for those LIN-items that belong to a given region. From row 1070 we find, using pointer PLIST, that the first LIST-item of the region EGGENBERG is in row 1071. Each LIST-item consists of:

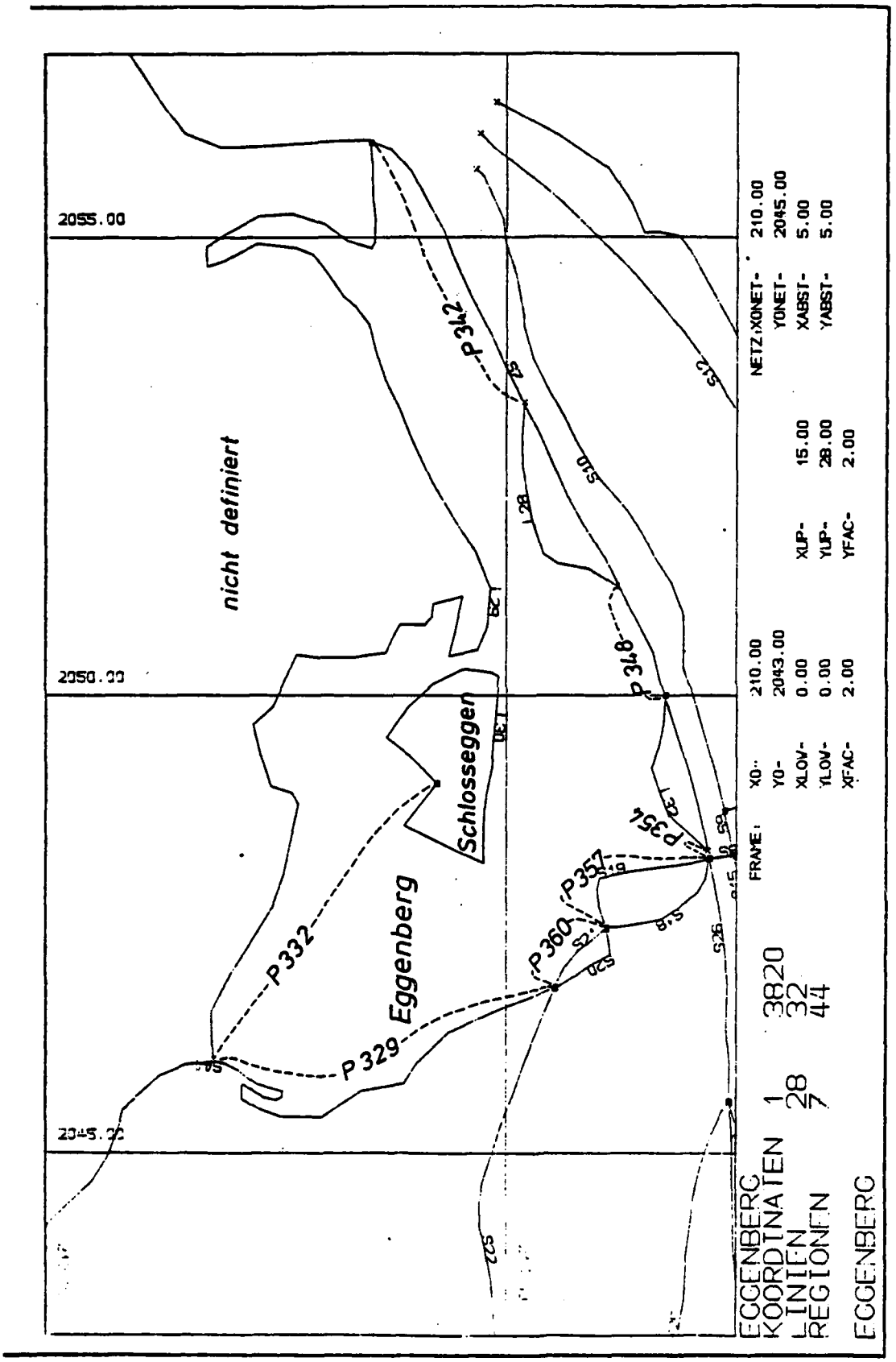


Figure 3.4(a): Section of map Graz around Eggenberg.

```
REG-Items        ( RA          PLIST )
          LIST-Items     ( PL          PNEXT )
                                  LIN-Items     (    -              LA )
                                                (   PR1            PR2 )
    328 :          197          331
                                   329 :       262143          262142
EGGENBERG                          330 :          328            1181
          331 :          329          334
                                   332 :       262143          262142
                                   333 :          328             328
          334 :          332          337
                                   335 :       262143              30
                                   336 :          328             363
          337 :          335          338
          338 :          332          341
                                   339 :       262143              29
                                   340 :          328               0
          341 :          339          344
                                   342 :       262143          262142
                                   343 :          328            1070
          344 :          342          347
                                   345 :       262143              28
                                   346 :          328             370
          347 :          345          350
                                   348 :       262143          262142
                                   349 :          328            1070
          350 :          348          353
                                   351 :       262143              32
                                   352 :          328             365
          353 :          351          356
                                   354 :       262143          262142
                                   355 :          328            1070
          356 :          354          359
                                   357 :       262143          262142
                                   358 :          328            1232
          359 :          357          362
                                   360 :       262143          262142
                                   361 :          328            1222
          362 :          360            0
    363 :          198          364
SCHLOSSEGGEN  364 :      335            0
    1181 :          44          1182
S00000000045  1182 :   262140         1185
                                  1183 :       262143          262142
                                  1184 :         1181               0
          1185 :         1183         1188
                                  1186 :       262143          262142
                                  1187 :         1181               0
          1188 :         1186         1189
          1189 :       262141         1192
                                  1190 :       262143          262142
                                  1191 :         1181               0
          1192 :         1190         1193
          1193 :          329            0
```

Figure 3.4(b): Sections of the graph around region 197, Eggenberg.

Row number: PL PNEXT.

Each pointer PL points to the next LIN-item or designates the start or end of the contour of a region. PNEXT points to the row with the next LIST-item.

3.2.5 Illustration of the graph using an area.

Figures 3.4(a), (b) show two areas, one enclosed by the other. The pseudo-line has special significance in this case that is typified by an island in a lake: non-connected parts of the boundary-polygon (contour) exist in the form of the edges of the lake and island. The pseudo-line connects both and must be passed twice.

From this and the illustration of the skeleton in fig.3.3 it is now straight forward to follow the section of the graph for the areas EGGENBERG and SCHLOSSEGGEN.

Row 328 contains the REG-item of region EGGENBERG with pointer PLIST pointing to row 331 with the first LIST-item. This in turn has pointer 329 to the (first) LIN-item in row 329. Among the LIST-items there are two rows with pointer PL pointing to the same LIN-item in row 332: this designates a pseudo-line (LA equals 262142) separating area 328 from 328 or more appropriately passing through to area 328.

Region EGGENBERG is delimited by region SCHLOSSEGGEN, by skeleton S44 and by the undefined background (S44 internally has the name S00000000045) . This is the reason why figure 3.4(b) also contains the graph sections for these regions.

3.2.6 Line- and region files

Line- and region files form the connection between the graph and the lowest level consisting of the sequence of points in the coordinate list. Line- and region files contain descriptive, thematic and statistical information concerning a line or region. An element of these files is composed as follows:

(a) Name ..... 12 characters. During data acquisition or tagging each line and skeleton automatically are assigned names. For lines this is L00000000XXX , for skeletons S00000000XXX , where 'XXX' represents successive running numbers. Names of regions can be freely selected by the user in the process of building up the graph.

(b) Windows ..... 4 values. The first two represent the smallest x,y coordinates of a line or skeleton, the last two the dimension in x- and y-direction of the data bank coordinate system.

(c) Pointer ..... points to the respective item in the graph.

(d) Coordinates ..... 2 values delimiting the range of rows in the coordinate list for the polygon of a line or skeleton.

(e) Starting point ... of a line or skeleton.

(f) End point ..... of a line or skeleton. These points permit rapid compilation of an area's boundary or contour without directly addressing the coordinate list. Areas do not have start- and end points, these exist only for lines or skeletons. With areas the values for start- and end points are filled by zeroes.

(g) 15 more positions . for statistical measures, for example the type of a region. In our specific application we define:

0 undefined
1 water (not skeleton!)
2 river, canal (only skeleton!)
3 built-up area
4 range land , agriculture
5 forest
6 rock
7 road (skeleton)
8 rail

Figure 3.5 presents an example of a printout of 3 parts of a line- and region file.

3.2.7 Coordinate list

This file contains point coordinates forming polygonal lines. Each point can be addressed by his position in the sequential file. Figure 3.6 shows an example of a printout of the coordinate list for lines L00000000031, L00000000032, L00000000033.

3.2.8 Coordinate list of the boundary of an area

We use an intermediary data structure before and after geometrical transformation of polygons. Polygons here are closed boundaries of areas. The structure is sequential. Two areas where one is totally enclosed by the other are represented as follows:

(i) Each line segment is preceded by a pair of values marking the start- and ending indices of the sequence of points of the line segment.

a)

*************** LINIEN VON 28 BIS 32 ***

NR	NAME	FENSTER			ZEIGER		KOORDINATEN		STARTPUNKT		ENDPUNKT	
28	L0000000029	215.17	2051.19	1.02	2.01	345	488	496	216.20	2051.19	215.19	2053.20
29	L0000000030	211.75	2046.03	3.09	10.00	339	497	554	213.53	2056.32	211.83	2046.03
30	L0000000031	213.72	2048.16	1.22	2.12	335	555	572	214.24	2049.04	214.24	2049.04
31	L0000000032	209.16	2047.44	.72	.87	0	573	588	209.71	2047.45	209.71	2047.44
32	L0000000033	216.57	2048.30	.61	1.70	351	589	597	216.73	2050.30	217.18	2048.30

b)

*************** REGIONEN VON 36 BIS 44 ***

NR	NAME	FENSTER			ZEIGER		KOORDINATEN		STARTPUNKT		ENDPUNKT	
36	S0000000037	223.06	2041.63	.44	1.32	962	1597	1600	223.06	2042.95	223.50	2041.63
37	S0000000038	219.92	2041.61	3.60	3.49	978	1601	1613	223.52	2041.61	219.92	2045.04
38	S0000000039	219.93	2035.76	3.61	9.26	1247	1614	1641	219.93	2045.32	223.54	2035.76
39	S0000000040	221.26	2034.82	2.26	.97	1302	1642	1647	223.52	2035.79	221.26	2034.82
40	S0000000041	223.53	2035.77	2.15	.27	855	1648	1654	223.53	2035.77	225.68	2036.04
41	S0000000042	223.52	2036.05	2.18	5.56	868	1655	1670	225.69	2036.05	223.52	2041.61
42	S0000000043	223.54	2040.61	2.21	1.00	879	1671	1676	223.54	2041.60	225.75	2040.61
43	S0000000044	223.05	2040.61	2.68	2.34	967	1677	1685	225.73	2040.61	223.05	2042.96
44	S0000000045	206.43	2043.43	9.11	3.29	1181	1686	1731	215.54	2046.82	206.43	2044.86

c)

*************** REGIONEN VON 196 BIS 200 ***

NR	NAME	FENSTER			ZEIGER		KOORDINATEN		STARTPUNKT		ENDPUNKT	
196	BAHNHOFGUERT	213.85	2055.03	1.08	1.02	309	0	0	.00	.30	.00	.00
TYP 3												
197	EGGENBERG	211.75	2046.03	5.44	10.00	328	0	0	.00	.30	.00	.00
TYP 3												
198	SCHLOSSEGGEN	213.70	2048.16	1.22	2.12	363	0	0	.00	.30	.00	.00
TYP 4												
199	BAHNHOF	214.57	2048.30	.61	1.70	365	0	0	.00	.30	.00	.00
TYP 8												
200	WIENERSTR	215.17	2051.19	1.02	2.01	370	0	0	.00	.30	.00	.00
TYP 3												

Figure 3.5: Section of a printout of line- and region files.

```
**    KOORDINATEN  VON    555 BIS    597    ********** ********** ****
                555 : (   214.2430 , 2049.0370 )
                556 : (   213.9550 , 2049.3360 )
                557 : (   213.6980 , 2049.5410 )
                558 : (   213.9400 , 2049.8760 )
                559 : (   214.2060 , 2050.1180 )
                560 : (   214.5280 , 2050.2830 )
                561 : (   214.8450 , 2050.2450 )
                562 : (   214.9220 , 2050.2070 )
                563 : (   214.8920 , 2049.7930 )
                564 : (   214.8610 , 2049.4960 )
                565 : (   214.8430 , 2049.2110 )   L00000000031
                566 : (   214.7840 , 2048.9080 )
                567 : (   214.7640 , 2048.7200 )
                568 : (   214.7490 , 2048.4290 )
                569 : (   214.7540 , 2048.1630 )
                570 : (   214.4560 , 2048.2990 )
                571 : (   213.8930 , 2048.5740 )
                572 : (   214.2430 , 2049.0370 )
                573 : (   209.7110 , 2047.4450 )
                574 : (   209.8830 , 2047.4860 )
                575 : (   209.8620 , 2047.6510 )
                576 : (   209.7540 , 2047.8210 )
                577 : (   209.7130 , 2048.0260 )
                578 : (   209.5830 , 2048.2060 )
                579 : (   209.4510 , 2048.3090 )
                580 : (   209.2720 , 2048.1820 )
                581 : (   209.2380 , 2048.0730 )
                582 : (   209.1600 , 2047.9780 )   L00000000032
                583 : (   209.2530 , 2047.9020 )
                584 : (   209.3580 , 2048.0130 )
                585 : (   209.4470 , 2048.0200 )
                586 : (   209.5570 , 2047.7770 )
                587 : (   209.6290 , 2047.5760 )
                588 : (   209.7090 , 2047.4370 )
                589 : (   216.7260 , 2049.9990 )
                590 : (   216.7040 , 2049.6830 )
                591 : (   216.6560 , 2049.4380 )
                592 : (   216.5710 , 2049.2040 )
                593 : (   216.6600 , 2048.9440 )   L00000000033
                594 : (   216.7330 , 2048.7590 )
                595 : (   216.8100 , 2048.6520 )
                596 : (   216.9730 , 2048.4960 )
                597 : (   217.1820 , 2048.3040 )
```

Figure 3.6: Example of a printout for coordinate list of map Graz (compare figure 3.5).

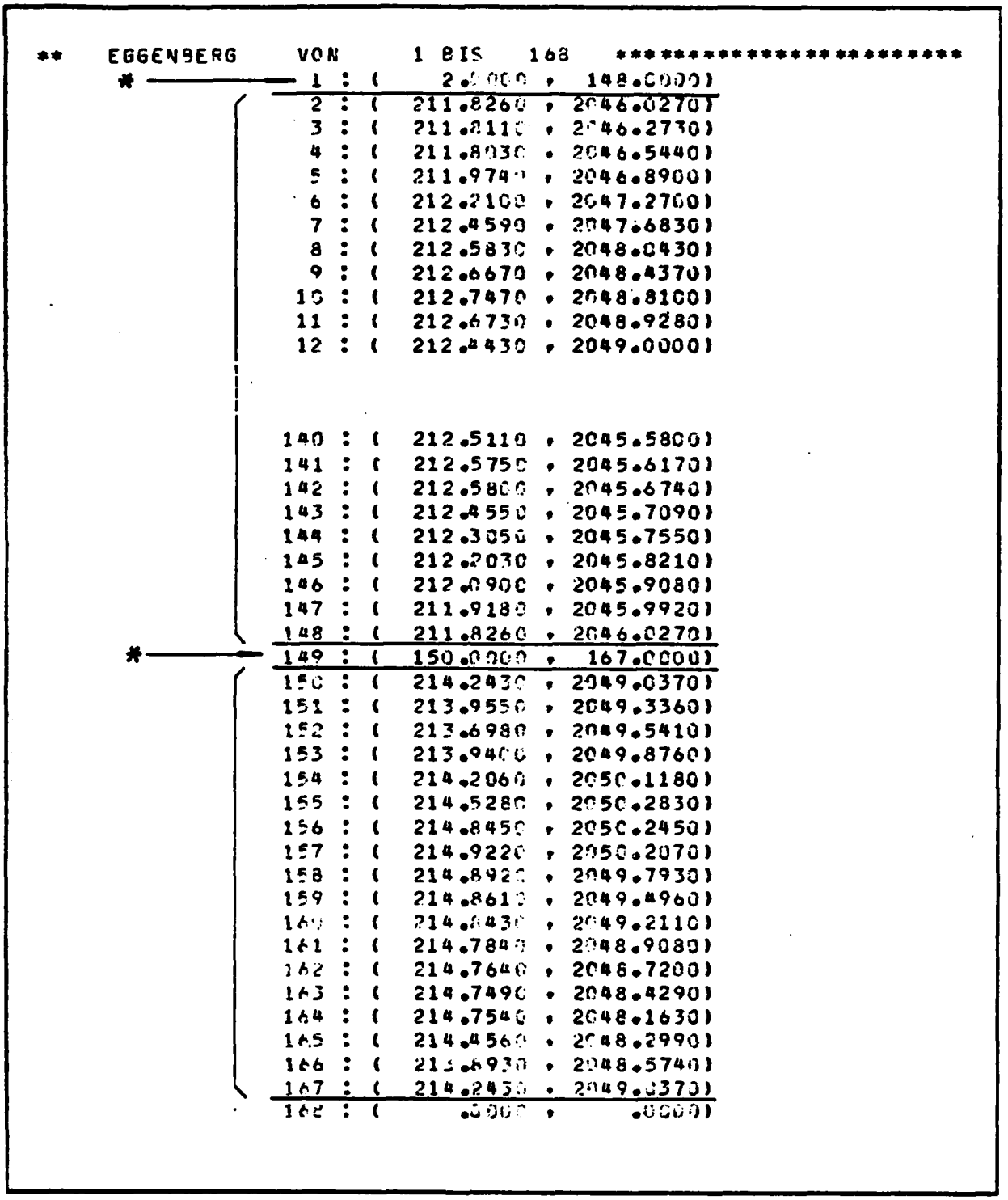


Figure 3.7: Coordinate list of a polygon describing area Eggenberg. (*) marks value pairs preceding a section of a polygon.

(ii) These line segments are written sequentially into the coordinate list.

Figure 3.7 illustrates such a coordinate list with the preceding pairs of values here as indices 1 and 49.

3.2.9 Binary matrix (raster)

For logical operations with graphical data, such as operations of intersection, union and taking the negative the raster format is well suited. Therefore a provision exists in the data bank to convert areas for which a boundary polygon exists to a raster format. The appropriate data structure is a binary matrix. Coordinates are defined by a window preceding the matrix. The window definition is the same as that in the line- and region file. It is followed by a 2-dimensional matrix M:

$$M(i,j) = 1 \quad \text{if } (i,j) \in \text{area}$$
$$M(i,j) = 0 \quad \text{elsewhere.}$$

The elements of the matrix are ordered row by row in a compact format: the last bit of one row is directly followed by the first bit of the next row. This is currently implemented to form arbitrary rectangular binary matrices with 72 000 elements.

Figure 3.8 shows region EGGENBERG as a binary matrix.

3.3 ALGORITHMS FOR DATA INPUT AND OUTPUT

The current implementation of computer programs is to serve in the main task of merging map and image data. Limitations of resources has therefore mainly affected the user ergonomy in data acquisition and editing. To further illustrate the current implementation stage we outline the data input and output. Actual applications software now exists for merging maps and images. This is described in a separate chapter 4.

3.3.1 Data Acquisition

(a) Step 1: Digitizing of points, lines and skeletons.

We currently do off-line digitization on a manual digitizing table[1]) and then transfer the data to a disk.

---

[1]) This is possible due to the cooperation of the Agrartechnische Abteilung of the Government of Styria.

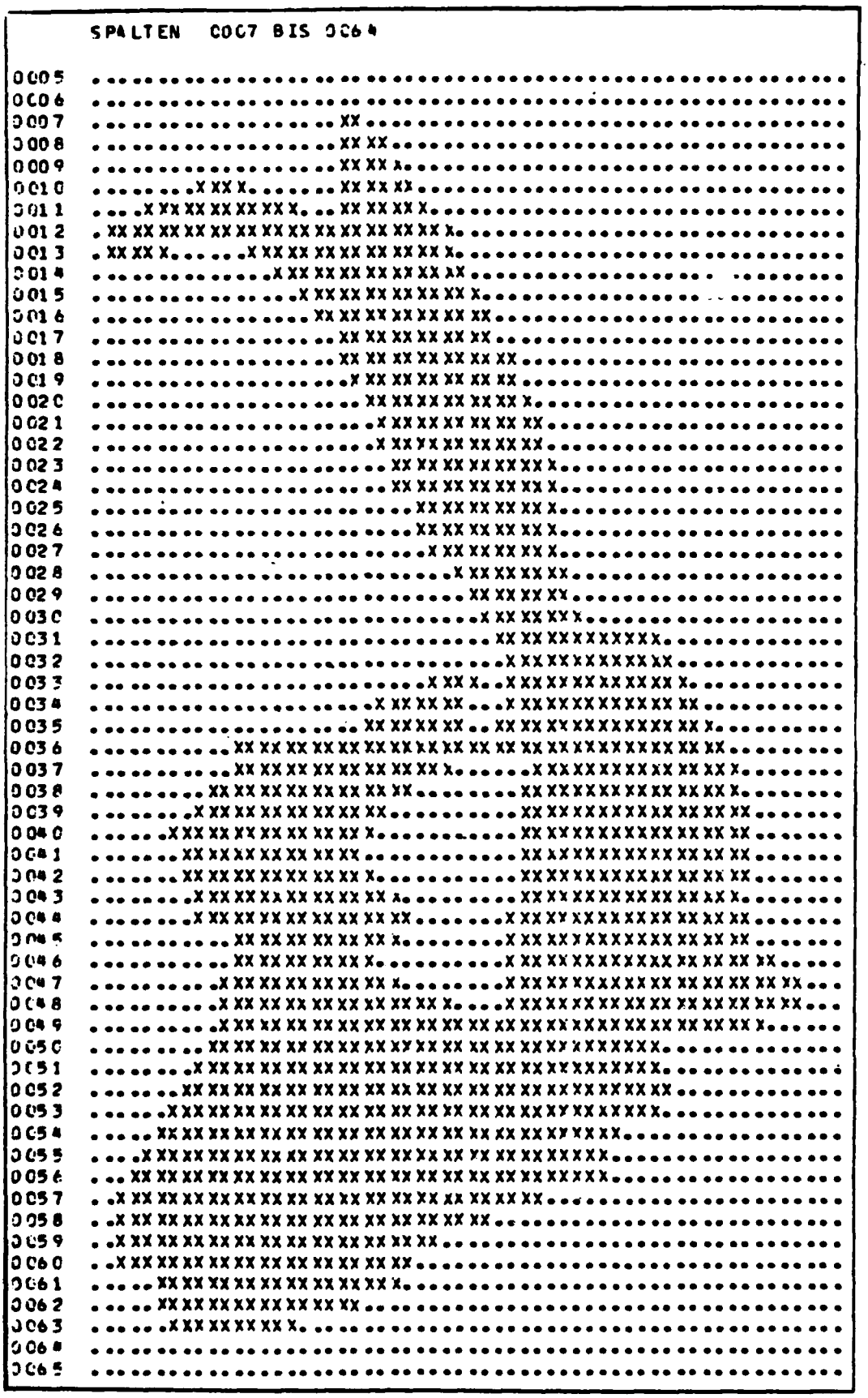


Figure 3.8: Printout of binary matrix of area EGGENBERG.

Editing is with a text-editor to preparate the data for establishment of the coordinate list, line file and the skeleton part of the region file. This is done from the raw digitizer data in program MAPREAD. This provides the following elements in the line and region files.

Name = 'S' or 'L' with automatically assigned number;

Window with automatically defined dimensions;

Indices (addresses) of start- and endpoint in the coordinate list;

Coordinates of start- and endpoint.

For quality control and to support the 2nd step the data can be output in the form of lists or graphical plots (program MAPOUT, see section 3.3.2).

(b) Step 2: Graph generation

Program GRAPHREAD generates the graph, fills up the region file by adding to it all areas, and completes the line- and region files with all pointers.

Input to program GRAPHREAD at this time is manual. It consists of names and types of regions and identification of area boundaries as a sequence of numbers of lines and pseudolines. Limited resources prevented us from going into automation of graph generation.

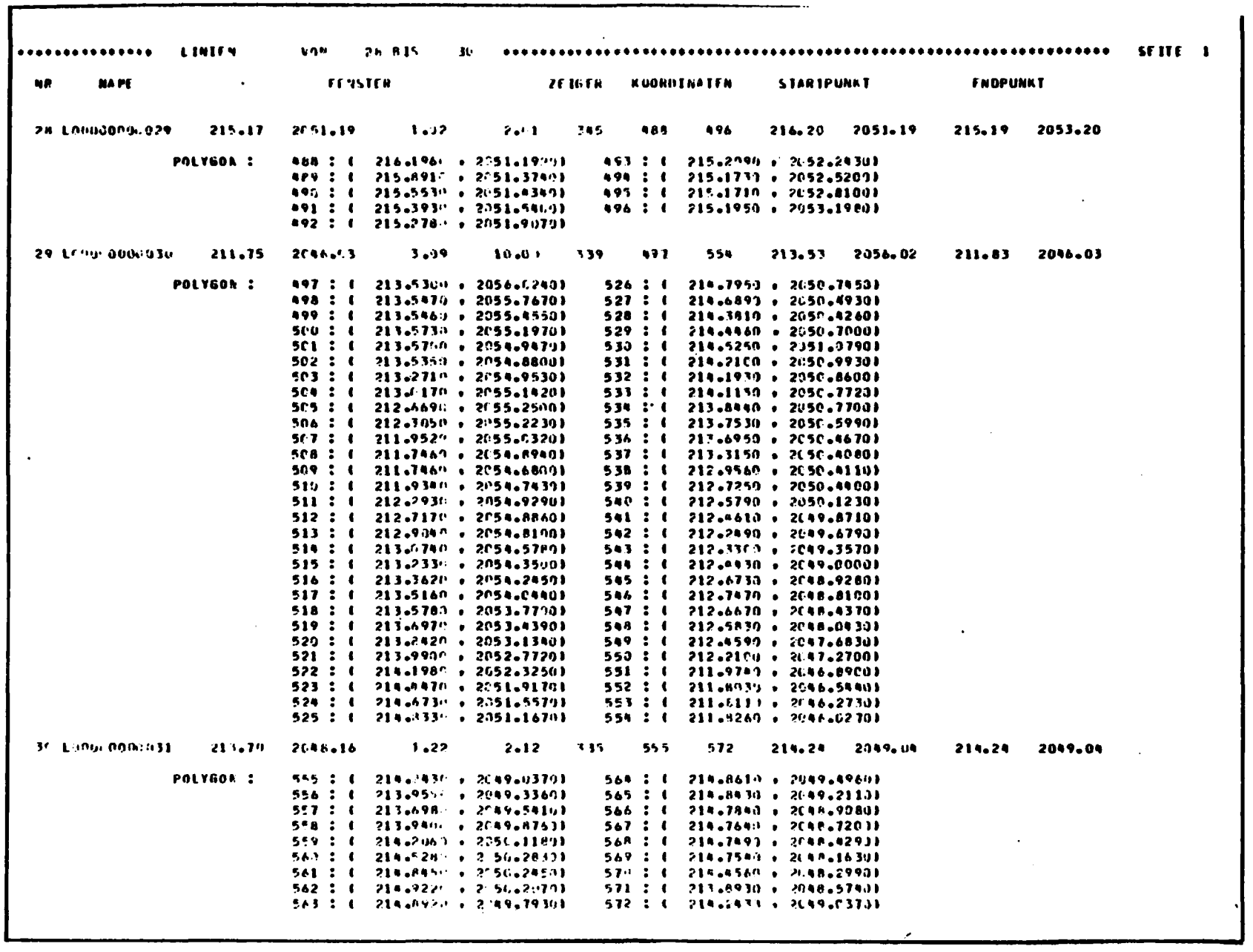
(c) Printout for control

The file sizes can be easily verified in a specific control printout. Figure 3.9 presents a section of such a control printout of the data bank GRAZ. Units are computer words (36 bits) as occupied on disk. The logical unit of the coordinate file is a pair of coordinate values occupying 2 words.Therefore there are 1910 coordinate pairs in the file. The logical uniut of the region- and line files in 28 words. There are thus 6216:28 = 222 regions and 3304:28 = 118 lines.

```
KARTENDATENBANK DIBAG-MAPS          :GRAZ


KARTE :GRAZ          FILE :KOORDINATEN    IST    3820 WORTE LANG
KARTE :GRAZ          FILE :REGIONEN       IST    6216 WORTE LANG
KARTE :GRAZ          FILE :LINIEN         IST    3304 WORTE LANG
KARTE :GRAZ          FILE :STRUKTUR       IST    5000 WORTE LANG
```

Figure 3.9: Control printout of map data bank Graz.



```
**********  LINIEN      VON    26 BIS    30  **********                                                    SEITE  1

 NR    NAME              FENSTER                ZEIGER   KOORDINATEN    STARTPUNKT          ENDPUNKT

 28 L0000300%.029    215.17  2051.19    1.32     2.11    345    488    496   216.20  2051.19  215.19  2053.20

              POLYGON :   488 : (  216.1960 , 2051.1970)   493 : (  215.2090 , 2052.2430)
                          489 : (  215.8910 , 2051.3740)   494 : (  215.1730 , 2052.5200)
                          490 : (  215.5530 , 2051.4340)   495 : (  215.1710 , 2052.8100)
                          491 : (  215.3930 , 2051.5860)   496 : (  215.1950 , 2053.1980)
                          492 : (  215.2780 , 2051.9070)

 29 L0000000%030     211.75  2046.03    3.99    10.01    339    497    554   213.53  2056.02  211.83  2046.03

              POLYGON :   497 : (  213.5300 , 2056.0240)   526 : (  214.7950 , 2050.7450)
                          498 : (  213.5470 , 2055.7670)   527 : (  214.6890 , 2050.4930)
                          499 : (  213.5460 , 2055.4550)   528 : (  214.3810 , 2050.4260)
                          500 : (  213.5730 , 2055.1970)   529 : (  214.4460 , 2050.7000)
                          501 : (  213.5700 , 2054.9470)   530 : (  214.5250 , 2051.0790)
                          502 : (  213.5350 , 2054.8800)   531 : (  214.2100 , 2050.9930)
                          503 : (  213.2710 , 2054.9530)   532 : (  214.1930 , 2050.8600)
                          504 : (  213.1170 , 2055.1420)   533 : (  214.1150 , 2050.7720)
                          505 : (  212.6690 , 2055.2500)   534 : (  213.8840 , 2050.7700)
                          506 : (  212.3050 , 2055.2230)   535 : (  213.7530 , 2050.5990)
                          507 : (  211.9520 , 2055.0320)   536 : (  213.6950 , 2050.4670)
                          508 : (  211.7460 , 2054.8940)   537 : (  213.3150 , 2050.4080)
                          509 : (  211.7460 , 2054.6800)   538 : (  212.9560 , 2050.4110)
                          510 : (  211.9380 , 2054.7430)   539 : (  212.7250 , 2050.4000)
                          511 : (  212.2930 , 2054.9290)   540 : (  212.5790 , 2050.1230)
                          512 : (  212.7170 , 2054.8860)   541 : (  212.4610 , 2049.8710)
                          513 : (  212.9040 , 2054.8100)   542 : (  212.2490 , 2049.6790)
                          514 : (  213.0740 , 2054.5780)   543 : (  212.3300 , 2049.3570)
                          515 : (  213.2330 , 2054.3500)   544 : (  212.0930 , 2049.0000)
                          516 : (  213.3620 , 2054.2450)   545 : (  212.6730 , 2048.9280)
                          517 : (  213.5160 , 2054.0480)   546 : (  212.7470 , 2048.8100)
                          518 : (  213.5780 , 2053.7700)   547 : (  212.6670 , 2048.4370)
                          519 : (  213.6970 , 2053.4390)   548 : (  212.5830 , 2048.0830)
                          520 : (  213.2420 , 2053.1340)   549 : (  212.4590 , 2047.6830)
                          521 : (  213.9900 , 2052.7720)   550 : (  212.2100 , 2047.2700)
                          522 : (  214.1980 , 2052.3250)   551 : (  211.9780 , 2046.8900)
                          523 : (  214.0470 , 2051.9170)   552 : (  211.8030 , 2046.5480)
                          524 : (  214.6730 , 2051.5570)   553 : (  211.6110 , 2046.2730)
                          525 : (  214.8330 , 2051.1670)   554 : (  211.4260 , 2046.0270)

 30 L0000000%031     213.79  2048.16    1.22     2.12    335    555    572   214.28  2049.08  214.28  2049.08

              POLYGON :   555 : (  214.2430 , 2049.0370)   564 : (  214.8610 , 2049.4960)
                          556 : (  213.9550 , 2049.3360)   565 : (  214.8830 , 2049.2110)
                          557 : (  213.6980 , 2049.5810)   566 : (  214.7840 , 2048.9080)
                          558 : (  213.9400 , 2049.8763)   567 : (  214.7640 , 2048.7200)
                          559 : (  214.2060 , 2050.1180)   568 : (  214.7490 , 2048.4290)
                          560 : (  214.5280 , 2 50.2840)   569 : (  214.7580 , 2048.1630)
                          561 : (  214.8450 , 2 50.2450)   570 : (  214.4560 , 2048.2990)
                          562 : (  214.9220 , 2 50.2070)   571 : (  214.8930 , 2048.5740)
                          563 : (  214.8920 , 2 49.7930)   572 : (  214.1431 , 2049.0370)
```

Figure 3.10: Printout (section) of line file of GRAZ completed by coordinate file.

3.3.2 Data output

Data output is with program MAPOUT in the form of lists and graphical plots. Output routines are interactively started (format free input with input requests). Output devices available at this time are line printers and drum or flat bed plotters.

Program MAPOUT was used to generate figures 3.2 through 3.7 and 3.9. It can also produce combinations of lists, such as of coordinate and line lists as shown in figure 3.10. Graphical plots in figure 3.3(a) were enlargements of specified features of the data bank of which a larger portion was presented as figure 3.2, all plotted with the same program MAPOUT.

3.4 CONCLUSION

Several structures exist for the digital map data in the current application. These can be grouped into three main groups:

- the map data bank with its coordinate files, element lists and topological structure;
- the polygon (vector) structure of individual map features;
- the (binary) matrix structure for compatibility with digital images.

The apparent complexity of a digitally implemented topological structure has great benefits when the data must be analyzed and lengthy search processes can be drastically shortened.

The polygon (vector) structure of individual map features is an intermediate structure that is feature-oriented. It has the drawback of wasted storage if large data quantities need to be kept in this format. But in our application this drawback is irrelevant because the structure is only used as an intermediate one for the ultimate presentation in a binary image: this is the structure needed to input map data into a digital image processing system.

4. MAP DATA IN IMAGE COMPATIBLE FORM

4.1 GENERAL

The main current application of the map data bank of chapter 3 is to serve for experiments to automatically merge an image with a map. Applications programs therefore address the problem of extracting data from the data bank in a format suitable to use with digital images.

Within the overall flow of operation this task occurs in program MAPTRANSFORM which in turn is called in statement (9) of procedure OVERLAY (figure 2.5).

The application programs can be grouped according to the data structure used. figure 4.1 presents an overview.

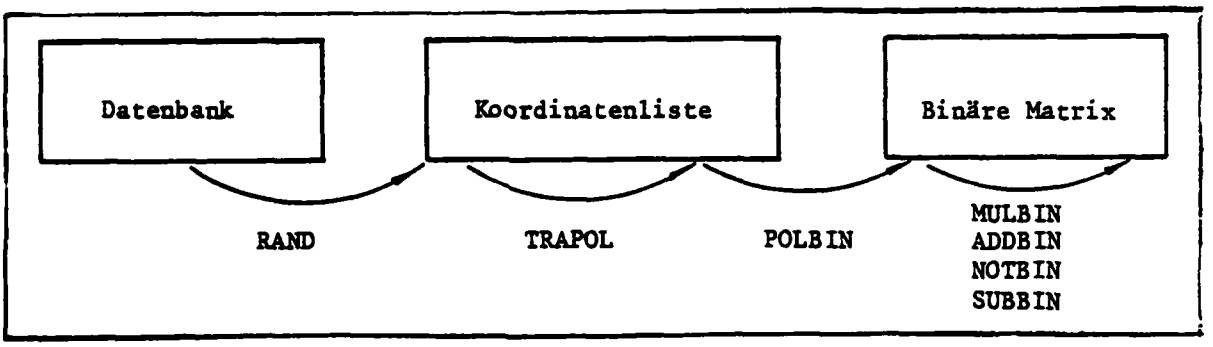


Figure 4.1: Existing application programs for the map data bank.

Programs RAND and POLBIN serve to convert one data structure into another one. The other programs are to operate on a given data structure: TRAPOL performs a geometrical transformation, set operations are feasible with programs MULBIN, ADDBIN, NOTBIN, SUBBIN; program MASS computes the centroid of a feature from either its coordinate list or its binary matrix. One may argue that MASS creates a new data structure (the center of gravity) from other structures.

Program RAND extracts from the data bank the contour of a feature or object in the form of a coordinate list. These data are then available in either a vector format or in raster presentation for further processing in the context of digital image analysis.

4.2 EXTRACTION OF A FEATURE FROM THE DATA BANK, PROGRAMM RAND

As was mentioned before it is through routine MAPTRANSFORM that a map feature is transformed into an image compatible form.

The PDL formulation is presented in figure 4.2. It calls the programs listed in figure 4.1.

```
(1)procedure MAPTRANSFORM (map, object, binarymatrix);
(2)begin
(3)      RAND (map, object, polygon 1);
(4)      TRAPOL (polygon 1, polygon 2);
(5)      POLBIN (polygon 2, binarymatrix);
(6)end
```

Figure 4.2: PDL formulation of program MAPTRANSFORM

Program RAND is important for ARSIM because it addresses individual features of the map data bank. Given the name (regnam) of an object RAND generates the boundary polygon as a stream of x,y coordinates.

The PDL-description of RAND is presented in figure 4.3. RAND first uses program KASUCH (map, regnam) to search in the region file for the number NREG of the description of the object and from there takes the REG-item associated with object REGNAM. Should the description of the region show that it is a skeleton, then the search is simple; a file is filled with a sequence of skeleton points from the skeleton coordinate list (Fig.3.1) in positive and negative direction (printl(+1), printl(-1)). This ensures that in a conversion to a binary matrix one has 2 border points along each direc - tion of a raster line. This also enables one later to widen the skeleton.

Is the object an area then RAND goes along all LIST- and LIN-items of the region using pointer IREG and routine CONTINUOUSLINE. The main subroutine in RAND is CONTINUOUSLINE. It is presented in figure 4.4.

In order to describe RAND and CONTINUOUSLINE, let us now consider two different cases, first a simple one, with an area that is only defined by lines, then the complex one, with an area within another area, defined by lines and skeletons.

4.2.1 Area defined only by lines

The search in the region file results in a pointer IPL (= element (c) in chapter 3.2.6) to a REG-itm (IREG). This in turn produces the first LIST-item ('liste') and enters into subroutine CONTINUOUSLINE. This defines through routine REGLIS, whether or not the associated LIST/LIN-item concerns a line

```
(1)     procedure RAND (map, regnam, polygon);
(2)     begin
(3)         LPSØ:=Ø;
(4)         LISTA:=Ø;
(5)         NREG:=KASUCH (map, regnam);
(6)         if NREG <Ø then "not found"
(7)                    else
(8)         begin
(9)             readrg (S1, NREG);
(10)            IREG:=IPL (S1);
(11)            if RA (IREG) ≠ NREG then "DB-error"
(12)                                 else
(13)            begin
(14)                assign polygon file;
(15)                initiate polygon;
(16)                if regnam is a skeleton then
(17)                begin
(18)                    printl (+1);
(19)                    printl (-1);
(20)                    close polygon;
(21)                    close polygon file;
(22)                end                  else
(23)                begin
(24)                    liste:= PLIST(IREG);
(25)                    while liste ≠ Ø do
(26)                    begin
(27)                        CONTINUOUSLINE (LPSØ, LISTA, liste);
(28)                        close polygon;
(29)                        liste:=LISTA;
(30)                        LPSØ:=LISTA:=Ø
(31)                    end
(32)                    close polygon file;
(33)                end
(34)            end
(35)        end
(36)    end.
```

Figure 4.3: PDL description of routine RAND.

```
(1)    procedure CONTINUOUSLINE (LPSØ,LISTA,liste);
(2)    begin
(3)        case REGLIS (liste) of
(4)        'L': begin
(5)                 read1(S1);
(6)                 case REGLIS(liste) of
(7)                 'L': begin
(8)                          read1(S2);
(9)                          switch:= 1;
(1o)                         if start 1=start 2 or start 1=end 2 then
(11)                         begin
(12)                             print1(-1);top:=end 1
(13)                         end                                    else
(14)                         begin
(15)                             top:=start 1; print1(+1)
(16)                         end
(17)                    end
(18)                'P' switch:= 2;
(19)                eof:begin print1(+1);switch:= 3 end;
(2o)            end
(21)       'P': begin
(22)                 readrg (S1);
(23)                 LPSØ: = line;
(24)                 case REGLIS (liste) of
(25)                 'L': begin
(26)                          read1 (S2);
(27)                          switch: = 1;
(28)                          top 1: = top: = NAEHER (start2,end2,S1)
(29)                     end
(3o)                 'P': switch: = 4;
(31)                 eof: begin print1(+1); switch:= 3 end;
(32)            end
(33)       eof: begin
(34)                 if LISTA = Ø then "no boundary";
(35)                 switch: = 3;
(36)            end
(37)       case switch of
(38)       1: begin
(39)              if top = end 2 then print1(-1)
(4o)                             else print1(+1);
(41)              L4;
(42)           end
(43)       2: begin
(44)              readrg (S2);
(45)              top:= NAEHER (start 1,end 1,S2);
(46)              if top=start 1 then print1(-1)
(47)                             else print1(+1);
(48)              L5;
(49)           end
```

Figure 4.4. Routine CONTINUOUSLINE(cont'd on nextpage).

```
(5o)        3:;
(51)        4: begin
(52)               if IREGA=IREGB then
(53)               begin
(54)                   case SPITZ of
(54)                   1: begin
(55)                          top 1:=top:=start 1;
(56)                          S2:=S1;
(57)                      end
(58)                   2: begin
(59)                          top 1:=top:=end 1;
(6o)                          S2:=S1;
(61)                      end
(62)                   3: begin
(63)                          LPSØ: = Ø;
(64)                          case REGLIS (liste) of
(65)                          'L': begin
(66)                                   read1 (S2);
(67)                                   if start 1=start2 or start 1=end 2 then
(68)                                   begin
(69)                                       print1(+1);print1(-1);top1:=start 1
(7o)                                   end                                  else
(71)                                   begin
(71)                                       top 1:=end1; print1(-1);print1(+1)
(72)                                   end;
(73)                                   if top=end 2 then print1(-1)
(74)                                                else print1(+1);
(75)                                   L4;
(76)                                   switch:=1;$ dont enter L5 in line loo $
(77)                               end
(78)                          'P': begin
(79)                                   readrg (S2);
(8o)                                   if start 1=start 2 or start 1=end 2 then
(81)                                   begin
(82)                                       print1(+1);print1(-1);top1:=start 1
(83)                                   end                                  else
(84)                                   begin
(85)                                       top 1:=end1;print1  (-1);print1(+1)
(86)                                   end;
(87)                                   IREGA:=IREGB
(88)                               end
(89)                          eof: begin "Ireg=skeleton";switch:= 3 end;
(9o)                      end
(91)               end             else
(92)               begin
(93)                   $ IREGA≠IREGB $
(94)                   readrg (S2);
(95)                   top:=start 1;
(96)                   if top≠start 2 & top≠end 2 then top:=end 1;
(97)                   top 1:= top;
(98)                   IREGA:=IREGB
(99)               end
(1oo)              if switch = 4 then L5;
(1o1)       end
(1o2)   end.
```

Figure 4.4. Routine CONTINUOUSLINE(continued).

```
(1)     procedure L4;
(2)     begin
(3)         while REGLIS (liste) = 'L' do
(4)         begin
(5)             readl (S1);
(6)             if top = endl then printl (-1)
(7)                           else printl (+1);
(8)         end
(9)         if REGLIS (liste) = 'P' then
(10)        begin
(11)            readrg(S2);
(12)            L5
(13)        end
(14)                                  else
(15)        if REGLIS (liste) = eof then L6;
(16)                                  else error;
(17)    end.
```

Figure 4.5: PDL-description of procedure L4 within CONTINUOUSLINE. L4 follows a contour along lines (segments).

('L'), a pseudo-line ('P') or the end of the feature ('eof'). With the present assumptions it must be a line 'L'. We have to find now the direction in which the sequence of coordinates must be ordered.

Therefore one reads from the line list the current line (readl (S1)) and the subsequent one (readl (S2)); S1 and S2 denote two different storage buffers. Then it is a trivial task to find the direction from statement (10) of program CONTINUOUSLINE. One now has to copy the points of the line from the coordinate file to the polygon file in the order resulting from statement (10), either as printl (+1) or printl (-1); "top" is set as the last point read and is kept for further use. Successive lines are then added on using procedure L4 (figure 4.5). This is repeated until pointer PNEXT of the LIST-item "liste" equals zero. Then procedure L6 is executed (figure 4.8) to terminate the polygon.

4.2.2 An area within another area defined by lines and skeletons.

This is the more complex case. It becomes relevant if a LIN-item represents a pseudo-line. One of the two pointers PR1, PR2 must then point to the parental REG-item, the other pointer can also point there. In that case one is dealing with a pseudo-line connecting for example one island with the boundary of a lake (area within area). Or the other pointer must point to a skeleton. Pseudo-lines connecting islands with a region boundary are detected in procedure REGLIS. Since they do not affect the polygon of the current boundary level the part of the line list between the first and the second appearence of such pseudo-lines are skipped in REGLIS and the type of the following LIN-item is returned. To resume the skipped list part the pointer to the first LIST-item after the pseudo-line is retained in LISTA. After completion of the current polygon the polygon circumscribing the island is added to the polygon file as shown in lines (25) to (31) of procedure RAND.

If a feature's contour meets with a skeleton the algorithm enters procedure L5 (Figure 4.6). The entry state is defined by the "top" of the contour generated so far and a skeleton stored in buffer S2. The direction in which the contour follows the skeleton is determined by the next LIST-item. There are 3 cases to distinguish.

Case SK1 (figure 4.7):

From the skeleton border the contour continues with a line (lines 49-55 of L5). Rename "top Ø". Procedure NAEHER determines the new top which is either the starting (start 1) or the end point (end 1) of the line so that it is nearer to the skeleton. Then the polygon along

```
(1)     procedure L5;
(2)     begin
(3)         LIND:= true;
(4)         while REGLIS(liste)= 'P' & LIND do
(5)         begin
(6)             if IREGA=IREGB then
(7)             begin
(8)                 case ISPITZ of
(9)                 1:begin
(1o)                      RASKEL(top,start 2,S2);
(11)                      top:=start 2
(12)                  end
(13)                2:begin
(14)                      RASKEL(top,end2,S2);
(15)                      top:=end 2
(16)                  end
(17)                3:begin
(18)                      if top =start 2 then
(19)                      begin
(2o)                          RASKEL(top,end2,S2);
(21)                          RASKEL(end2,top,S2);
(22)                      end              else
(23)                      begin
(24)                          RASKEL(top,start2,S2);
(25)                          RASKEL(start2,top,S2);
(26)                      end;
(27)                      LIND:=false
(28)                  end
(29)            end             else
```

Figure 4.6.:PDL-description of procedure L5 within CONTINUOUSLINE. L5 follows a contour along pseudolines,which separate the region from adjacent skeletons (continued on next page).

```
(3o)            begin ∮ IREGA ≠ IREGB ∮
(31)                 readrg(S1);
(32)                 if start 1= start 2 or start 2=end 1 then
(33)                 begin
(34)                     RASKEL(top,start2,S2);
(35)                     top:=start 2
(36)                 end                                    else
(37)                 begin
(38)                     RASKEL(top,end2,S2);
(39)                     top:=end2
(4o)                 end
(41)                 IREGA: = IREGB;
(42)                 S2:=S1
(43)            end
(44)        end while;
(45)        if not LIND then L4
(46)                    else
(47)            if REGLIS (liste)= 'L' then
(48)            begin
(49)                 readl (S1);
(5o)                 top ∅:= top;
(51)                 top:=NAEHER(start 1, end1, S2);
(52)                 RASKEL(top ∅, top, S2);
(53)                 if top=start 1 then printl(+1)
(54)                                else printl(-1);
(55)                 L4
(56)            end                      else
(57)                 if REGLIS(liste) = eof then L7
(58)                                        else   error
(59)    end
```

Figure 4.6: PDL-description of procedure L5 within CONTINOUS-LINE. L5 follows a contour along pseudolines, which separate the region from adjacent skeletons (continued from previous page).

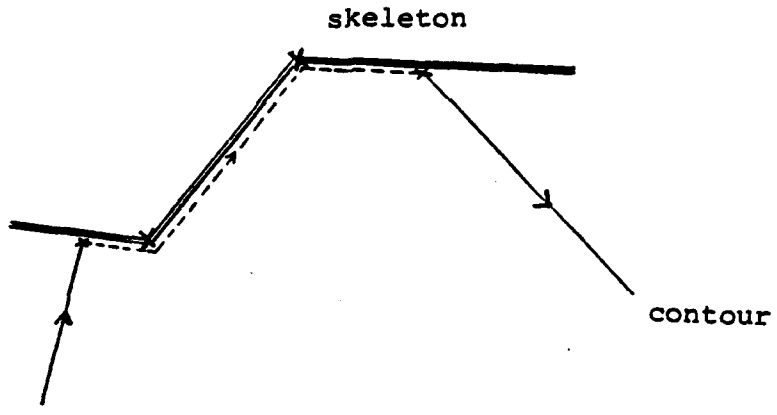


Figure 4.7: Case SK1 of a skeleton meeting a contour.

the skeleton between top Ø and top is generated by procedure RASKEL. From top the algorithm runs as described in chapter 4.2.1 (procedure L4).

```
(1)     procedure L6;
(2)     begin
(3)         if LPSØ ≠ Ø  then
(4)         begin
(5)             IREGA: = PR1 (LPSØ);
(6)             if IREG = IREGA then IREGA = PR2 (LPSØ);
(7)             readrg (S1);
(8)             RASKEL(top,top1,S1);
(9)         end;
(10)    end·
```

Figure 4.8: PDL-description of procedure L6 within CONTINUOUS-LINE. L6 closes a contour when coming from L4.

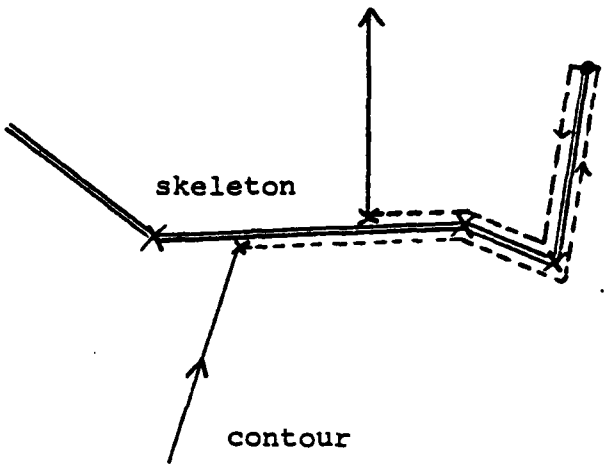


Figure 4.9: Case SK2 of a skeleton meeting a contour

Case SK2 (figure 4.9):

It follows a pseudo line of the same skeleton (IREGA= IREGB; lines 6-29 of L5). This happens for example when a dead end road enters the region. The contour must follow the skeleton until its limit and return on the other side of the skeleton. The two limits of a skeleton are marked in the LIST by special PL-pointers: 26214Ø marks a start and 262141 an endpoint. Function ISPITZ determines which limit enters the region. It returns 1 if it is a start point, 2 if it is an end point and 3 in a special case: The contour joins the skeleton in a limit point and the other limit lies inside the region. This skeleton is described by exactly 2 pseudolines and the two limit marks. Therefore the contour must follow the skeleton to the opposite limit and return back to the starting point, which remains the top of the contour polygon. Hence the algorithm returns to procedure L4. In the first two cases ISPITZ = 1 and ISPITZ = 2 the way of the contour can be traced until the limit point, which becomes the new top. The algorithm is then in the same state than before and iterates in procedure L5.

```
(1)     procedure L7;
(2)     begin
(3)         if LPSØ=Ø then RASKEL(top,top 1,S2)
(4)                   else
(5)         begin
(6)             IREGA:=PR1(LPSØ);
(7)             if IREG=IREGA then IREGA:=PR2(LPSØ);
(8)             if IREGA=IREGB then
(9)                   case ISPITZ of
(1o)                  1:begin
(11)                        RASKEL(top,start2,S2);
(12)                        RASKEL(start2,top 1,S2)
(13)                  end
(14)                  2:begin
(15)                        RASKEL(top,end2,S2);
(16)                        RASKEL(end2,top 1,S2)
(17)                  end
(18)                  3:if top=start2 then
(19)                  begin
(2o)                        RASKEL(top,end2,S2);
(21)                        RASKEL(end2,top 1,S2)
(22)                  end             else
(23)                  begin
(24)                        RASKEL(top,start2,S2);
(25)                        RASKEL(start2,top 1,S2)
(26)                  end
(27)                        else
(28)            begin ß IREGA≠IREGBß;
(29)                  readrg (S1);
(3o)                  topØ:=start 1;
(31)                  if end 1=start 1 or end 1= end 2 then
(32)                    topØ:=end 1;
(33)                  RASKEL(top,topØ,S2);
(34)                  RASKEL(topØ,top 1,S1)
(35)            end
(36)        end
(37)    end.
```

Figure 4.10: PDL-discription of procedure L7 within CONTINUOUS-LINE. L7 closes a contour when coming from L5.

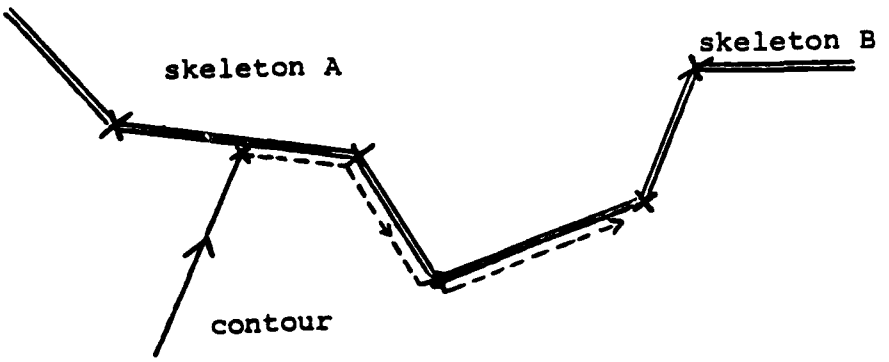


Figure 4.11: Case SK3 of a skeleton meeting a contour.

Case SK3 (figure 4.11):

It follows a pseudo-line of another skeleton (IREGA=IREGB; lines 3Ø - 43 of L5). The contour then follows the first skeleton (by RASKEL) until the joint point with the other one, which is the new top to continue and interate in L5.

There exists a case where the end of the LIST follows the skeleton; this is not treated here separately, because a contour is always a closed line. The first LIST-item therefore follows the last one and allows to distinguish the three cases. Some detail that is different here from the previous one is shown in procedure L7 (figure 4.10).

4.2.3 Calling program RAND

Figure 4.12 is an example of an interactive call to program RAND with the response as an operator receives it. Two areas are called:

EGGENBERG and SCHLOSSEGGEN

that we have used in chapter 3.

```
** $COM      BEGRENZUNGSPOLYGONE BESTIMMEN

** $RAND GRAZ.EGGENBERG
 FENSTER VON EGGENBERG   =    211.746   2066.027      5.436      9.997
 KOORDINATENFILE EGGENBERG    ERSTELLT.

ELAPSED:     3.000  TOTAL:     2.584  CPU:      .100  I/O:     1.828  SEC


** $RAND SCHLOSSEGGEN
 FENSTER VON SCHLOSSEGGEN=    213.698   2048.167      1.224      2.120
 KOORDINATENFILE SCHLOSSEGGEN ERSTELLT.

ELAPSED:      .773  TOTAL:     1.859  CPU:      .029  I/O:     1.202  SEC
```

Figure 4.12: Call to extract regions EGGENBERG and SCHLOSS-EGGEN from the data bank GRAZ.

The call includes the name of the data bank (GRAZ) and of the region (e.g. EGGENBERG).Interactive commands are preceded by a '$' sign. The printout also generates '**'. The machine response has no preceding symbol. Times given in the last row of figure 4.12 are machine dependent. On our machine (UNIVAC 1100/81) the CPU-time was 0.029 seconds.

4.3 OPERATIONS WITH THE EXTRACTED FEATURE

4.3.1 TRAPOL

Procedure TRAPOL (figure 4.13) performs a simple linear geometric transformation of the polygon created by program RAND. It also can widen individual skeletons to a specified width.

The transformation T(x) is:

$$\underline{T}(\underline{x}) = \underline{A}\,\underline{x} + \underline{b}$$

with vectors $\underline{T}(\underline{x})$, $\underline{x}$ and $\underline{b} \in R^2$ and a square transformation matrix $\underline{A}$. Transformation parameters must be computed in a separate program.

```
(1)   procedure TRAPOL (polygon 1, polygon 2);
(2)   begin
(3)       read parameters of transformation T;
(4)       assign  file for polygon 2;
(5)       initiate file;
(6)       for all part of polygon 1 do
(7)       begin
(8)           for all point p of part do
(9)               polygon 2: = polygon 2 + T (p)
(10)          close polygon;
(11)      end
(12)      close file;
(13)  end.
```

Remarks to procedure TRAPOL:

1) Description of a polygon as a syntagmatic grammar (VAUQUOIS, 1970):

   $G = (\mathbb{R}^2 \cup \{\text{endofpart}\}, \{\text{polygon, part, point}\}, \text{polygon}, P)$

   with

   P = {polygon $\longrightarrow$ part polygon / part

   part $\longrightarrow$ point part endofpart /endofpart

   point $\longrightarrow$ $(x,y) \in \mathbb{R}^2$}

2) "polygon + point" means attaching the point.

3) "close polygon" writes the endofpart.

4) $\underline{T}\ (\underline{p}) = \underline{A}\ \underline{p} + \underline{b}$ with

$$\underline{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \quad \text{and}$$

$$\underline{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Figure 4.13: Procedure TRAPOL

Widening a skeleton is achieved by displacing each part of the polygon in a positive direction for one half of the required width. Polygon elements are then intersected and the intersection points are then used as the new polygon points.

```
** $COM  3 ENTSPRECHUNGSPUNKTE BESTIMMEN DIE TRANSFORMATIONSFUNKTION

** $MASS 206. 2040.   206. 2057.   220. 2040.  .8
**          1.   40.8    1.    1.    60.   40.8
TX =       4.214*X +        .000*Y +    -867.143
TY =        .000*X +      -2.341*Y +    4816.800

** $COM   BEGRENZUNGSPOLYGONE INS NEUE KOORDINATENSYSTEM TRANSFORMIEREN:

** $TRAPOL EGGENBERG.EGPOL
 TRANSFORMIERTER FILE EGPOL          HAT   168 PUNKTE
    SEIN FENSTER :     25     3     25    27

ELAPSED:     .554  TOTAL:    1.083  CPU:     .015  I/O:     .453  SEC
```

Figure 4.14: Call of linear transformation of region EGGENBERG.

```
** $TRAPOL S[illegible]S45.S45POL
 TRANSFORMIERTER FILE S45POL         HAT    93 PUNKTE
    SEIN FENSTER :      2    24     41    10

ELAPSED:     .314  TOTAL:     .952  CPU:     .016  I/O:     .326  SEC


** $COM    SKELETON VERBREITERN ( STRASSE 2 X 0.3 )

** $TRAPOL S45POL.S45POL-4, .3
 TRANSFORMIERTER FILE S45POL-4       HAT   102 PUNKTE
    SEIN FENSTER :      2    24     41    10

ELAPSED:     .3[illegible]  TOTAL:     .85[illegible]  CPU:     .[illegible]  I/O:     .22[illegible]  SEC
```

Figure 4.15: Call of program TRAPOL to transform and widen the skeleton.

Figure 4.14 is an example on an interactive call to the linear transformation. Here, transformation parameters are computed using 3 control points in an interactive call to program MASS that is then followed by TRAPOL. MASS uses 3 points (6 coordinates) in the map system, followed by 3 points (6 coordinates) in the transformed system. The call to TRAPOL requires two coordinate lists: the one to be transformed and the one after transformation (EGPOL).

Figure 4.15 is an example for transforming skeleton S000000000045 into S45POL, and then widening it to 0.3 units and storing it in file S45POL-4.

4.3.2 POLBIN

A discussion of procedures is decribed by NEWMAN and SPROULL (1979). The procedure used in our case is illustrated in Figure 4.16. One intersects each part of the polygon with equidistant (vertical) straight lines. Intersection points we ordered along the partial polygon (horizontally), and are then also ordered within each straight (vertical) line. One then has a raster image of the contour.

```
(1)     procedure POLBIN (polygon, binmatrix)
(2)     begin
(3)         binmatrix: = (Ø);
(4)         for all part of polygon do
(5)         begin
(6)             binloc: = (Ø);
(7)             PØ: = first point of part;
(8)             for all point P1 ∈ part - {PØ} do
(9)             begin
(10)                if  x(P1) - x(PØ) = Ø then
(11)                   for i:=y(PØ) until y(P1) do
                              binloc (x(PØ),i):= 1;
(12)                                  else
(13)                   INSORT (store, PØ, P1);
(14)                PØ: = P1;
(15)            end
(16)            for all columns IX of binloc do
(17)                while ∃ ya, ye ∈ store do
(18)                   for iy: =  [ya+.5]  until  [ye+.5]  do
(19)                       binloc (IX, IY): = 1;
(20)            binmatrix: = | binmatrix - binloc | ;
(21)        end
(22)    end.
```

Figure 4.16: Procedure POLBIN.

It is possible to number all edge raster points on a straight line and set all raster points between odd-numbered and even-numbered given edge raster points to 1, all others to 0. This then gives a binary matrix of the area, a so-called "fill-in".

```
** $COM   BEGRENZUNGSPOLYGONE IN BINAERE MATRIZEN UMFORMEN:

** $POLBIN EGPOL,EGBIN,IW
 AUS EGPOL        EGBIN          MIT FENSTER      1      1     56     35 GENERIERT

ELAPSED:      .992  TOTAL:     1.075  CPU:       .061  I/O:       .391  SEC
```

Figure 4.17: Call to program POLBIN to convert vector to raster data.

Figure 4.17 is a sample call to program POLBIN. The binary matrix is to be stored in file EGBIN. IW is a window of 56 by 35 pixels. If IW is not given, then POLBIN itself computes the minimal window to fit the binary image. The binary result is shown in Figure 4.18.

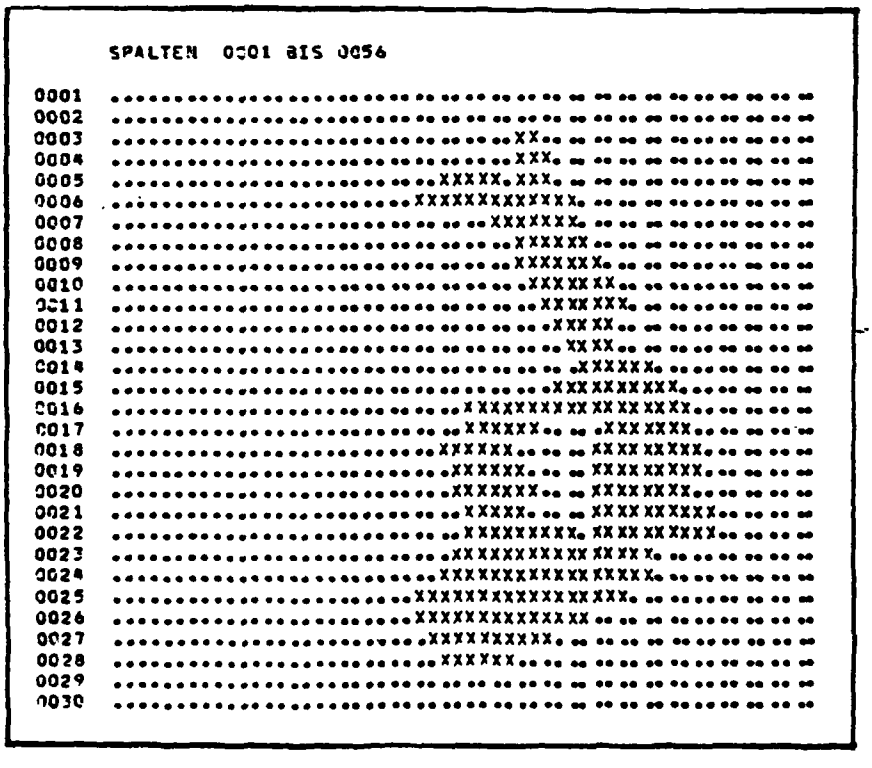


Figure 4.18: Example of binary matrix EGGENBERG.

4.3.3 Geometric set-operations

The raster format is well suited for the most important geometric set-operations. Two or more binary images can be connected on a bit level using logical functions that are very fast: the currently used computer has 36 bits per word and 36 logical operations can be done in parallel. Program MULBIN forms the intersection of two areas (logical operator AND), ADDBIN forms the union (logical operator OR), NOTBIN forms the negative of an area (logical operator NOT) and SUBBIN produces the absolute difference between two areas (EXCLUSIVE OR).

Figure 4.19 shows the printout of the program call and the result of a union of region EGGENBERG in binary format (stored as SCHBIN) and a road (S45BIN). Programs LIBIN and STOBIN are input/output routines.

4.3.4 Map information as an image

The binary matrices now enable one to generate chloroplethmaps of the regions in the data bank. Be M_i, $i = 1, \ldots, n$ mutually exclusive binary matrices and g_i, $i = 1, \ldots, n$ $[n \in 0,255]$ associated "gray values" we then can build a gray-tone digital image B:

$$B = \sum_{i=1}^{n} (g_i \cdot M_i)$$

Image B can now be processed in an image processing system such as our DIBAG (LEBERL, KROPATSCH, 1979).

An example of such a chloropleth presentation is shown in figure 4.20 with 4 regions:

areas EGGENBERG, SCHLOSSEGGEN, BAHNHOF, and skeletons S000000000088, S00000000045.

4.3.5 Program MASS

MASS computes the center of gravity S of a region and its area for both vector or raster format. The formulae are:

(a) vector format with polygon (x_i, y_i), $i = 1, \ldots n$

```
a)
** $COM   VEREINIGUNG VON SCHLOSS MIT S45 :


** $LIBIN SCHBIN
BINAERE MATRIX VON FILE SCHBIN      MIT FENSTER     1     1    56    35


** $ADDBIN S45BIN


** $STOBIN VEREINIGUNG
 BINAERE MATRIX AUF FILE VEREINIGUNG  GESPEICHERT


b)
      SPALTEN  0001 BIS 0056

0001  ..........................................................
0002  ..........................................................
0003  ..........................................................
0004  ..........................................................
0005  ..........................................................
0006  ..........................................................
0007  ..........................................................
0008  ..........................................................
0009  ..........................................................
0010  ..........................................................
0011  ..........................................................
0012  ..........................................................
0013  ..........................................................
0014  ..........................................................
0015  ..........................................................
0016  ..........................................................
0017  ..........................XXXXX.............................
0018  ........................XXXXXX............................
0019  ..........................XXXXX...........................
0020  ..........................XXXX............................
0021  ..........................XXXXX...........................
0022  ..........................X...............................
0023  ..........................................................
0024  ..........................................................
0025  ..........................XXXX............................
0026  ......................XX..XXXX............................
0027  ....................XXXXXXX..XXXX........................
0028  ....................XX...XXXXXXXX........................
0029  ..X.................XX..................................
0030  ..XXXXXX..........XX..................................
0031  ......XXXXXX.....XX..................................
0032  ........XXXXXXX.....................................
0033  ..........XXXXX......................................
0034  ..........................................................
0035  ..........................................................
```

Figure 4.19: Call to program ADDBIN to combine area SCHBIN with skeleton S45BIN (a road), and the result of extracting the program.

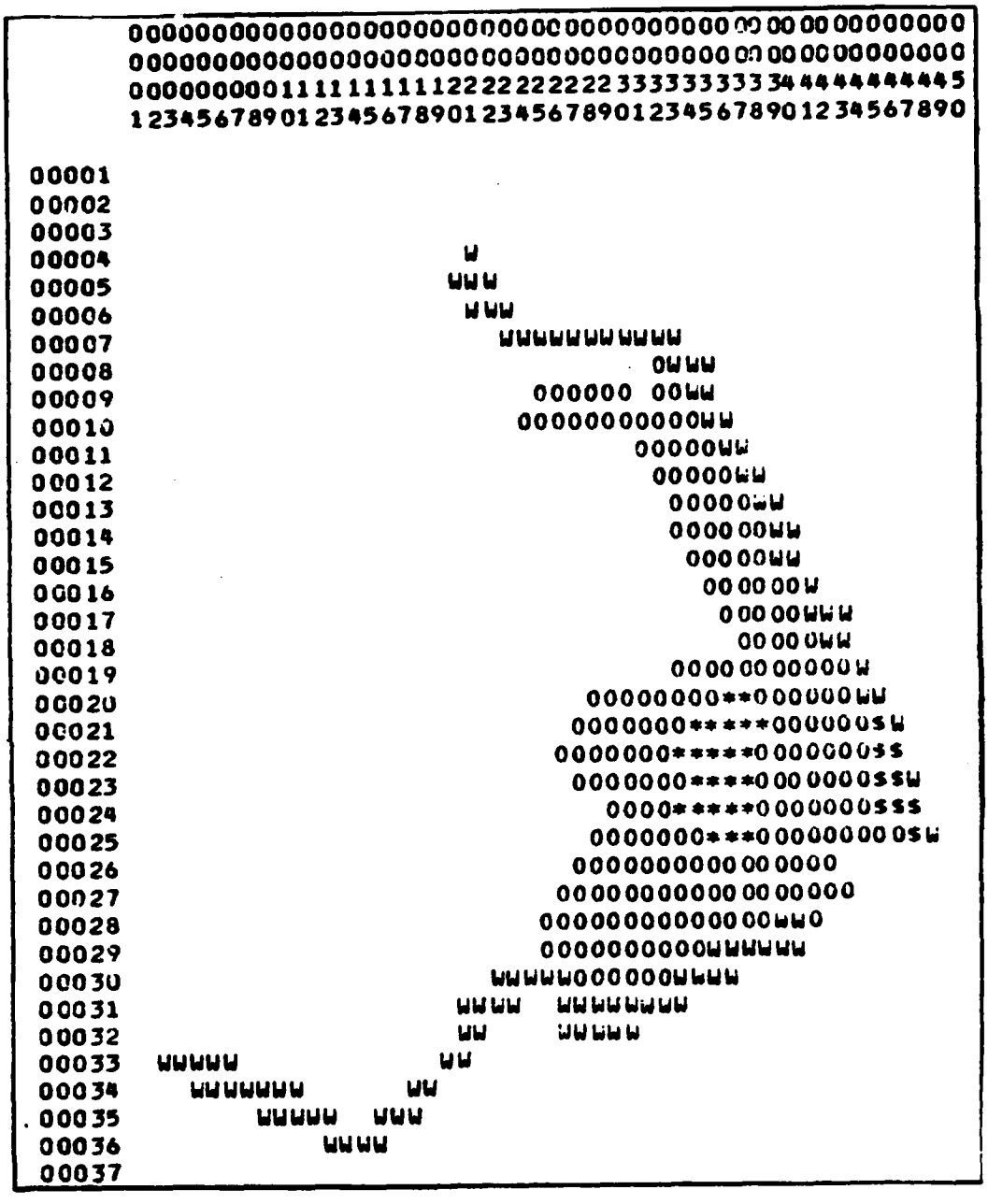


Figure 4.20: Chloropleth map of Eggenberg.

The following print symbols were used:

"$" ... Type 8, railway (BAHNHOF)
"W" ... Type 7, road (skeletons)
"O" ... Type 4, open area (EGGENBERG)
"*" ... Type 3, built up area (SCHLOSSEGGEN)

$$\text{area } ((x_i, y_i),\ i = 1, \ldots n) =$$

$$= \frac{1}{2} \sum_{i=1}^{n} (y_i \cdot x_{i+1} - y_{i+1} \cdot x_i)$$

$$\text{with } x_{n+1} \equiv x_1 \ , \ y_{n+1} \equiv y_1$$

$$S_x \ ((x_i, y_i),\ i = 1, \ldots n) =$$

$$= \frac{1}{3} \cdot \frac{\sum_{i=1}^{n} ((y_i \cdot x_{i+1} - y_{i+1} \cdot x_i) \cdot (x_i + x_{i+1}))}{\sum_{i=1}^{n} (y_i \cdot x_{i+1} - y_{i+1} \cdot x_i)}$$

$$S_y \ ((x_i, y_i),\ i = 1, \ldots n) =$$

$$= \frac{1}{3} \cdot \frac{\sum_{i=1}^{n} ((x_i \cdot y_{i+1} - x_{i+1} \cdot y_i) \cdot (y_i + y_{i+1}))}{\sum_{i=1}^{n} (x_i \cdot y_{i+1} - x_{i+1} \cdot y_i)}$$

(b) binary matrix B (i,j)

$$\text{area } (\underline{B}) = \text{card } \{B(i,j) = 1\};$$

This is the number of matrix elements of B equal to 1.

$$S_x \ (\underline{B}) = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} (i \cdot B\ (i,j)) \right) / \text{area } (B)$$

$$S_y \ (\underline{B}) = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} (j \cdot B\ (i,j)) \right) / \text{area } (B)$$

Figure 4.21 is a sample call to program MASS to illustrate its result.

```
** $COM   SCHWERPUNKTE + FLAECHEN VON POLYGON UND BIN.MATRIX SCHLOSS :

** $LIBIN SCHBIN
BINAERE MATRIX VON FILE SCHBIN      MIT FENSTER     1     1    56    35 GEL

** $MASS SCHLOSSEGGEN.BIN
    SCHLOSSEGGEN: FLAECHE=       1.577 SCHWERPUNKT=      214.421    2049.326
    BINAERMATRIX: FLAECHE=      26.000 SCHWERPUNKT=       36.231      19.038
TX =       4.214*X +        .000*Y +    -867.402
TY =        .000*X +      -2.341*Y +    4816.871

** $COM   NEUER VERSCHIEBUNGSVEKTOR DURCH GLEICHSETZEN DER SCHWERPUNKTE
```

Figure 4.21: Call to program MASS to compute area and center of gravity, and corrects the transformation function.

Binary image SCHBIN is first read in. Then the call to MASS results in the area and centers of gravity of the vector presentation (file SCHLOSSEGGEN) and of the binary presentation (file SCHBIN). The results are in the coordinate system of the data bank (SCHLOSSEGGEN) and of the binary matrix (SCHBIN). We already had a transformation between the two shown as figure 4.13 with 3 manually selected control points. The center of gravity as obtained in figure 4.18 is now automatically transformed and if a discrepancy with the actual center of gravity in the other data structure exists then a shift is automatically applied to the transformation (compare transformation parameters of figures 4.13 and figure 4.18).

4.4 CONCLUSION

It has been shown how elements of a map data bank can be extracted and converted to a format compatible with images: this is through the generation of the contour of an object from the data bank and subsequent conversion to a binary image. This image of one object can be combined with those of other objects to a digital chloropeth presentation of the map contents. This can then be processed like an image: both the binary and the chloropleth presentation can be understood as synthetic digital images.

The binary presentation can also serve as a mask to define a specific section of an image. This application is equally important then that of serving as synthetic image.

The most complex searching must be performed in program RAND. However, due to the organisation of the map data bank with a separate planar graph, with region- and line files and a separate coordinate file, searching can be done efficiently and fast.

5. PRE-PROCESSING OPERATIONS WITH IMAGES

Any form of digital (or analog) image analysis benefits from an optimization of the image. This precedes the analysis and is denoted by "pre-processing". One may classify pre-processing activities as follows:

Single images:

- Restoration
- Enhancement and Coding
- Geometric Rectification

Multiple images:

- Data Compression
- Enhancement and Coding (e.g. ratioing, directional cosine, clustering, etc.)

Of the countless methods of image (pre-)processing there are some of particular significance in the current context. In order to define these we have to realize that features are extracted from a map data bank mainly in the form of contoures of an object; these are then presented in a binary (black and white only) form. It follows thus, that also the multispectral image should be converted to a compatible format. This can be achieved by a clustering operation of image pixels (region growing) and subsequent binary coding, to enhance specific areas or the contours and edges.

We conclude therefore that image data compression to a binary form, and for the detection of edges in an image , are essential pre-processing steps in our task.

We have therefore analyzed a number of methods to identify edges in images, and have implemented various data compression techniques in order to obtain a pre-processed image for subsequent line detection tasks. This chapter deals first with a mathematical definition of digital images, edges, lines and neighbourhoods. A number of pre-processing functions are then defined in mathematical terms and finally evaluated with experimental data.

5.1 DEFINITIONS

5.1.1 Digital image

We describe a digital image by a function:

$$g: \quad B \rightarrow G \qquad (5.1)$$

of a finite, k-dimensional image area $B \subset \mathbb{N}^k$, where $\mathbb{N}$ represents the natural numbers 1,2 ... and k is normally 2. The function maps B into a finite range of gray values $G \subset \mathbb{N}$. For multiple images, e.g. multisprectral ones, k = 3. We can define multiple images also conveniently by a vector function

$$g = (g_1, g_2, \ldots g_n)$$

$$g_i : B \rightarrow G_i$$

where i denotes the channels of a multiple image. Each element of the vector function can be represented by a matrix of gray values. These matrices consist of the pixels defined at each matrix location i (row), j (column). We may in the following denote a pixel as an image point $x \in B$ or by its location $(i,j) \in B$.

5.1.2 Neighbourhoods

We define on image area B a neighbourhood $\Gamma(x)$ to each image point x.

$\Gamma(x) \subset Z^k$ is the set of neighbours of $x \in B$ if

$$x \notin \Gamma(x) , \tag{5.2}$$

$$x \in \Gamma(y) \Longleftrightarrow y \in \Gamma(x) , \tag{5.3}$$

where Z is the set of integer numbers.

$\left\{ \Gamma^n(x) \subset B \mid n \in \mathbb{N} \right\}$ is a system of neighbourhoods around $x \in B$ if:

(a) $$\forall_{x \in B} \; \Gamma^{(o)}(x) := \{x\} \tag{5.4}$$

(b) $$\Gamma^1(x) := \Gamma(x) \cap B \tag{5.5}$$

(c) $$\Gamma^{n+1}(x) := \bigcup_{y \in \Gamma^n(x)} \Gamma(y) \cap B \tag{5.6}$$

It is to be noted that $\Gamma^n(x)$ does not necessarily contain $\Gamma^{n-1}(x)$. $\Gamma(x)$ may not need to be a subset of B while $\Gamma^1(x)$ is. $\Gamma(x)$ can have negative numbers, $\Gamma^1(x)$ not.

There exist various types of neighbourhoods as shown in figure 5.2, whereas figure 5.1 illustrates the Γ_4 neighbours of 0^{th}, 1^{st}, 2^{nd} and 3^{rd} order of a point x.

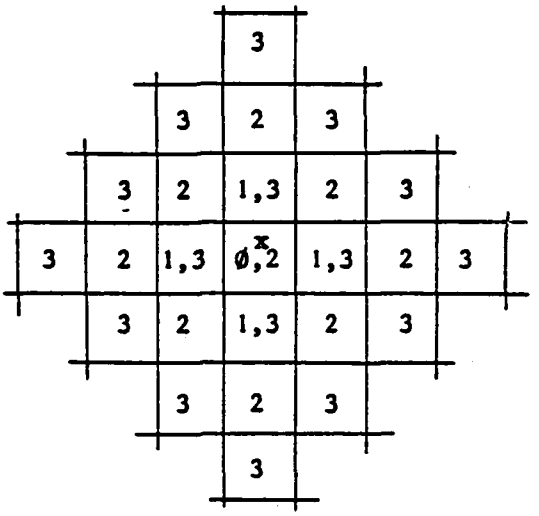


Figure 5.1: Γ_4-neighbourhoods $0^{th}, 1^{st}, 2^{nd}$ and 3^{rd} order of a point x.

5.1.3 Metric

We need for further use a metric in a digital image. All pairs of points $(x,y) \in B^2$ of an image area must be related through a neighbourhood of order n. A distance between x,y is then defined. We can prove the following

Lemma (1):

If there exists for each point $x \in B$ a system of neighbourhoods $\{\Gamma^i(x)\}$ and

$$\underset{x \in B}{\forall} \quad \underset{n \in \mathbb{N}_o}{\exists} \quad B = \bigcup_{i=\emptyset}^{n} \Gamma^i(x) \tag{5.7}$$

where $\mathbb{N}_o$ are the natural numbers and zero, then

$$d(x,y) := \min \left\{ k \in \mathbb{N}_o \;\middle|\; x \in \Gamma^k(y) \right\} \tag{5.8}$$

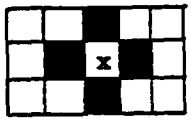
is a metric (distance) on B. A proof can be found in LEBERL and KROPATSCH (1978b).

For neighbourhoods Γ_4 and Γ_8 of Figure 5.2, it is easy to show that

$$d_4(\underline{x},\underline{y}) := \sum_{i=1}^{k} / x_i - y_i / \qquad (5.9)$$

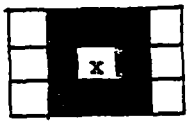
$$d_8(\underline{x},\underline{y}) := \max_{i=1,..k} (/x_i - y_i/) \qquad (5.10)$$

B 1) "4-Nachbar" (Rosenfeld, 1970):



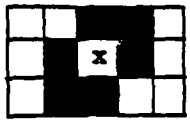
$\Gamma_4(x) := \{\blacksquare\}$ Beachte: card $(\Gamma_4(x) = 4$

B 2) "8-Nachbar" Königszug (Rosenfeld, 1970):



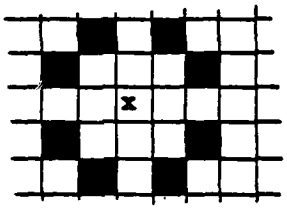
$\Gamma_8(x) := \{\blacksquare\}$ Beachte: card $(\Gamma_8(x) = 8$

B 3) "Schräge Nachbarmengen" könnten bei Bearbeitung geometrischer verzerrter Bilder sinnvoll sein:



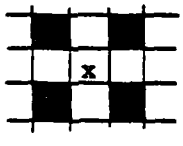
$\Gamma_v(x) := \{\blacksquare\}$

B 4) Als Beispiel einer besonderen Nachbarschaft sei die "Springer"-beziehung definiert:



$\Gamma_{sp}(x) := \{\blacksquare\}$

B 5) Den Zugmöglichkeiten beim "Dame"-Spiel entspricht:



$\Gamma_D(x) := \{\blacksquare\}$

Figure 5.2: Five examples of neighbourhoods in digital images.

5.1.4 Connectivity

We define a "path" to consist of image points p_i, $i = 1, \ldots n$, $p_i \in B$ and leading from p_1 to p_n if

$$\underset{1 \leq k < n}{\forall} \quad p_{k+1} \in \Gamma^1(p_k) \cup \{p_k\} \tag{5.11}$$

We define a subset M of B, $M \subset B$ to be "connected" if

$$\forall\, x,y \in M \;\exists \text{ path } (p_i) \text{ from } x \text{ to } y \text{ and } \forall_i\, p_i \in M$$

and a subset $M \subset B$ is "simply connected" if

- M is connected;
- $\overline{M} = B - M$ is connected, whereby in $\overline{M}$ a different neighbourhood relationship may be defined than in M.

5.1.5 Edges, lines, regions

We now have the tool to define edges, lines and regions.

(a) An "edge" e in a digital image is a pair $(i,o) \in B^2$ of neighbouring image points if

$$i \in \Gamma(o). \tag{5.12}$$

(b) The "weight" or "value" of an edge is a function

$$w: \quad B^2 \rightarrow \mathbb{R} \qquad (\mathbb{R} \text{ real numbers}). \tag{5.13}$$

(c) The direction of an edge is a function

$$\rho: \quad B^2 \rightarrow \mathbb{R} \tag{5.14}$$

We see that each pair of image points defines an edge. It is its weight that provides the edge with a physical meaning.

One needs to clearly differentiate between edges and lines. We therefore define a "line" L as a finite sequence of edges (e_i), $i = 1, n$ where consecutive edges e_i, e_{i+1} are neighbours of one another:

$$e_{i+1} \in \Lambda(e_i) \tag{5.15}$$

where symbol Λ denotes a neighbourhood on B^2, analogous to equations (5.2) and 5.3).

We can define open and closed lines: a line $L=(e_1,..e_n)$ is "closed" if

$$e_1 \in \Lambda(e_n).$$

It has length l where

$$l(L):=n \quad \text{if} \quad L = (e_i),\ i=1,\ \ldots\ n. \tag{5.16}$$

A "region" in a digital image is each simply connected subset of B.

A "contour" L of a region R is a line $L = (e_i)$, $i=1, \ldots n$ with edge $e_i = (p_i, q_i) \in B^2$ if

$$\underset{e_i=(p_i,q_i)}{\forall} \quad p_i \in R \ \&\ q_i \notin R \tag{5.17}$$

A region boundary consists of image points b_i, $i=1,\ldots,n$ and is defined if for all edges $(p_i, q_i) = e_i$ of a closed contour $L = (e_i)$, $i=1, n$ of region R one finds:

$$\underset{1\leq j\leq n}{\exists} \quad p_i = b_j \ \&\ j \leq i$$

5.1.6 Comment to definitions

It is significant to clearly define edges, lines and regions to avoid misunderstandings that can derive from common semantics. Definitions here are based on the concept of neighbourhood.

A possible confusion of the term "edge" may commonly be encountered with a situation as shown in figure 5.3.

range of gray values for R1 and R2

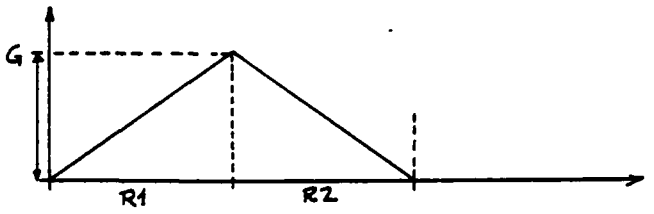


Figure 5.3: Roof and ridge-line.

With an edge-definition as the one of DAVIS (1973) one will have difficulty identifying an edge in figure 5.3. DAVIS (1973) defines:

"An edge is the boundary between two regions R_1, R_2 of different, constant gray values."

This could be written as:

$$(p,q) \in L \Longleftrightarrow p \in R_1 \;\&\; q \in R_2, \qquad R_1 \cap R_2 = \emptyset \tag{5.18}$$

If

$$\exists a<b<c \in G: \Big(\underset{x_1 \in R_1}{\forall} \; \underset{x_2 \in R_2}{\forall} \; a \le g(x_1) \le b \le g(x_2) \le c \Big) \vee \Big(\underset{x_1 \in R_1}{\forall} \; \underset{x_2 \in R_2}{\forall} \; a \le g(x_2) \le b \le g(x_1) \le c \Big) \tag{5.19}$$

then DAVIS (1973) must call all $(p,q) \in L$ "edges".

One can employ the definition of edges for multiple images. Let $B \subset \mathbb{N}^3$ with the neighbourhood:

$$\Gamma(x,y,k) := \left\{ (n,v,k) \;\middle|\; /n-x/ + /v-y/ = 1 \right\} \tag{5.20}$$

where K denotes the spectral channel.

Then the weight w of an edge e $((x_1,y_1,k_1),(x_2,y_2,k_2))$ is

$$w((x_1,y_1,k_1),(x_2,y_2,k_2)) := n^{-1} \sqrt{\sum_{i=1}^{n} (g(x_2,y_2,i) - g(x_1,y_1,i))^2} \tag{5.21}$$

We see that the weight of the edge is independent of image channel indices k_1, k_2.

5.2 EDGE DETECTION OPERATORS

5.2.1 General principles

A primary element of an edge operator is the number of pixels used to compute the value of an edge. There exists a great multitude of algorithms and numerous reviews (e.g. DAVIS, 1973; PRATT, 1978; LEBERL and KROPATSCH, 1978b). Of these methods, 4 are selected for detailed analysis: three are linear, one is non-linear.

A general formulation for an algorithm must include the case of multiple input images: $Image_k$, k = 1, ...n. Edge operations are performed in parallel: a "parallel" algorithm is defined to perform operations O_1, O_2, ...O_n where the outcome is not affected by the sequence of the operations.

```
procedure EDGE (Image, n, f, A)
begin
    for all i, j ∈ B  do
    A(i,j) = f(i,j,n,Image)
end.
```

Figure 5.4: PDL-description of an edge detection procedure.

Procedure EDGE addresses image points $b \in B$ and surroundings U^k of b of the order k:

$$U^k(b) := \left\{ x \in B \;\middle|\; x \in \Gamma^s(b),\ s \leq k \right\}$$

The operations of EDGE can therefore be called "local". As a result one obtains a new (gray-) value $g_1(b)$ for image point b:

$$g_1(b) := O_b\ (g(U^k(b))).$$

The parallel algorithm consists of performing operation O_b on all image points b. It is linear if:

$$\underset{x,y \in g(B)}{\forall} \quad \underset{a,b \in R}{\forall} \quad O(ax+by) = a \cdot O(x) + b \cdot O(y).$$

5.2.2 Linear algorithms

Typical linear parallel edge detection operations are: high-pass filtering, oriented numerical differentation, gradients, Laplace-operations. These methods can be described as an operation g_m with a mask m, that is a (local) function

$$m_x: \quad U(x) \rightarrow \mathbb{R}$$

on a surrounding U(x) of an image point $x \in B$. The value $m_x(b)$ of $b \in U(x)$ is only defined by the relative position of b with respect to x.

Operation g_m then follows as:

$$g_m(b) := \sum_{x \in U(b)} g(x) \cdot m_b(x).$$

Sample masks as used within an 8-neighbourhood Γ_8 of a 2-dimensional image area are:

(a)
$$\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$
horizontal forward difference on $U^1(x)$;

(b)
$$\begin{matrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{matrix}$$
Laplace-operator on $U^1(x)$

(c)
$$\begin{matrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{matrix}$$
double horizontal forward difference on $U^1(x)$.

(d) An iterative use of a type of Laplace-operator is implied in MONTOTO's (1977) method:

$m_b(b) = (\text{card } (U(b))-1)/\text{card } (U(b))$

$m_b(x) = -1 / \text{card } (U(b))$ if $x \neq b$.

5.2.3 Non-linear algorithms

Numerous authors have proposed non-linear edge detection methods (compare PRATT, 1978). These are usually more time-consuming than linear ones and therefore mostly limited to $U^1(x)$ of Γ_8.

5.2.4 Methods investigated

The following four methods were investigated:

(OP1) horizontal difference

f: $y_{ij} = / x_{ij} - x_{i+1j} /$

(OP2) Laplace operator

f: $y_{ij} = / 4x_{ij} - x_{i-1j} - x_{ij-1} - x_{i+1j} - x_{ij+1} /$

This corresponds to the mask in Γ_8:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

(OP3) operator of Rosenfeld (see DAVIS, 1973)

$$f: \quad y = \frac{1}{n} \left[\left| \sum_{k=1}^{n} (x_{i-k,j} - x_{i+k,j}) \right| + \left| \sum_{k=1}^{n} (x_{ij-k} - x_{ij+k}) \right| \right]$$

This operator has been implemented with n = 3. Fig.5.5 illustrates the image pixels entering into the operation for one output point.

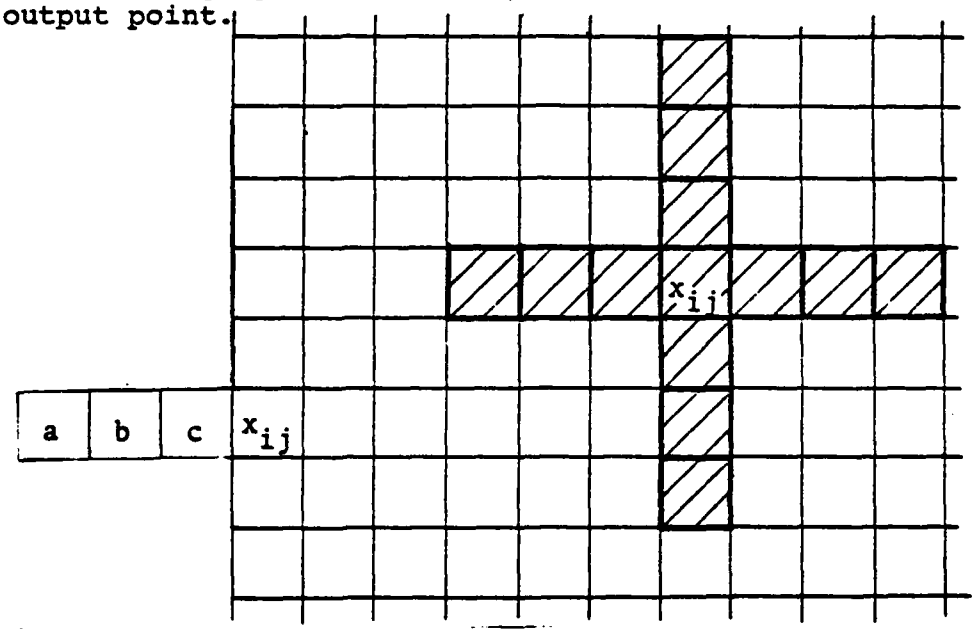


Figure 5.5: Pixel configuration used to compute the value of an edge with the operator of Rosenfeld for n = 3.

(OP4) operator of Roberts (PRATT, 1978)

$$f: \quad y_{ij} = \left[(x_{ij} - x_{i+1\ j+1})^2 + (x_{i+1,j} - x_{ij+1})^2 \right]^{1/2}$$

This is non-linear.

5.3 TEST DATA FOR EVALUATION OF EDGE OPERATORS

The six synthetic images of figure 5.6 were generated to test edge operators to define an optimum. The images all show simple geometric forms except for figure 5.6 (f) that is a segment of an actual LANDSAT image (lake).

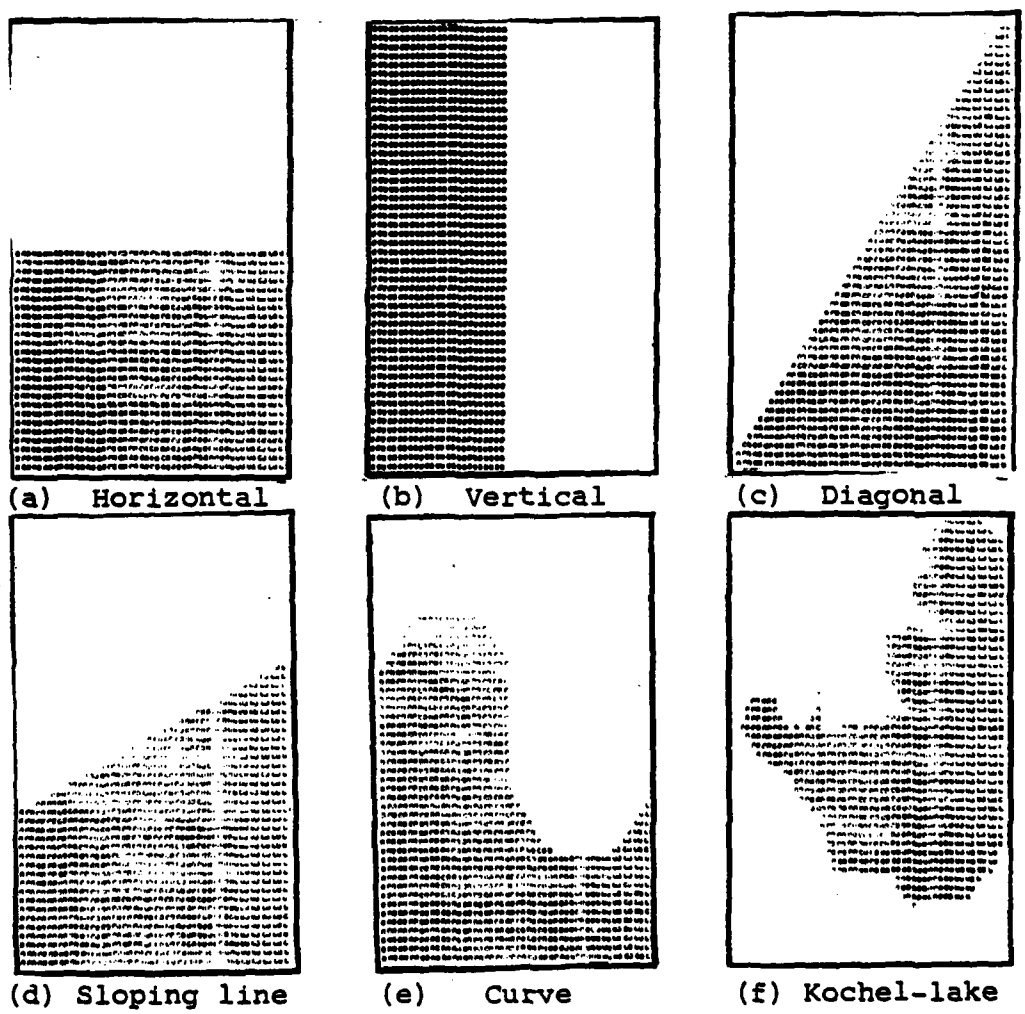


Figure 5.6: Six synthetic test images.

A theoretical edge reference image is defined assigning each image point $y_{ij} \in B$ a value:

$$g(y_{ij}) = \max_{k=1,8} (g(x_{ij}) - g(a_k))$$

with

$$a_k \in \Gamma_8 (x_{ij})$$

and x_{ij} are the pixels from the synthetic test image, gray values are 100 and 110. An example of an original image section and its edge reference image is shown in fig.5.7 (a),(b).

10	10	10	10	10	0	0	10	10	10
10	10	10	0	0	0	0	10	10	10
10	10	10	0	0	0	10	10	10	10
10	10	0	0	0	0	10	10	10	0
10	10	0	0	0	0	10	10	0	0

Figure 5.7: Section of synthetic image, bright areas with gray values 10, dark areas with 0.
(a) image; (b) edge image.

Similary edge reference images for figure 5.6 are shown in figure 5.8.

The test images were provided with noise:

noisy image = synthetic image + noise * f,

where noise has mean 0 and standard deviation 1; f is a noise factor. Noise added to images is shown in figure 5.9.

5.4 APPLICATION OF THE EDGE OPERATORS TO THE SYNTHETIC IMAGE

All four edge operators were applied to all synthetic images with noise factors 0,1,2,3,4.Then the result was evaluated as follows:

(a) thresholding of edge images creation of binary edge images;
thresholds used were 60,80,100,120,140;

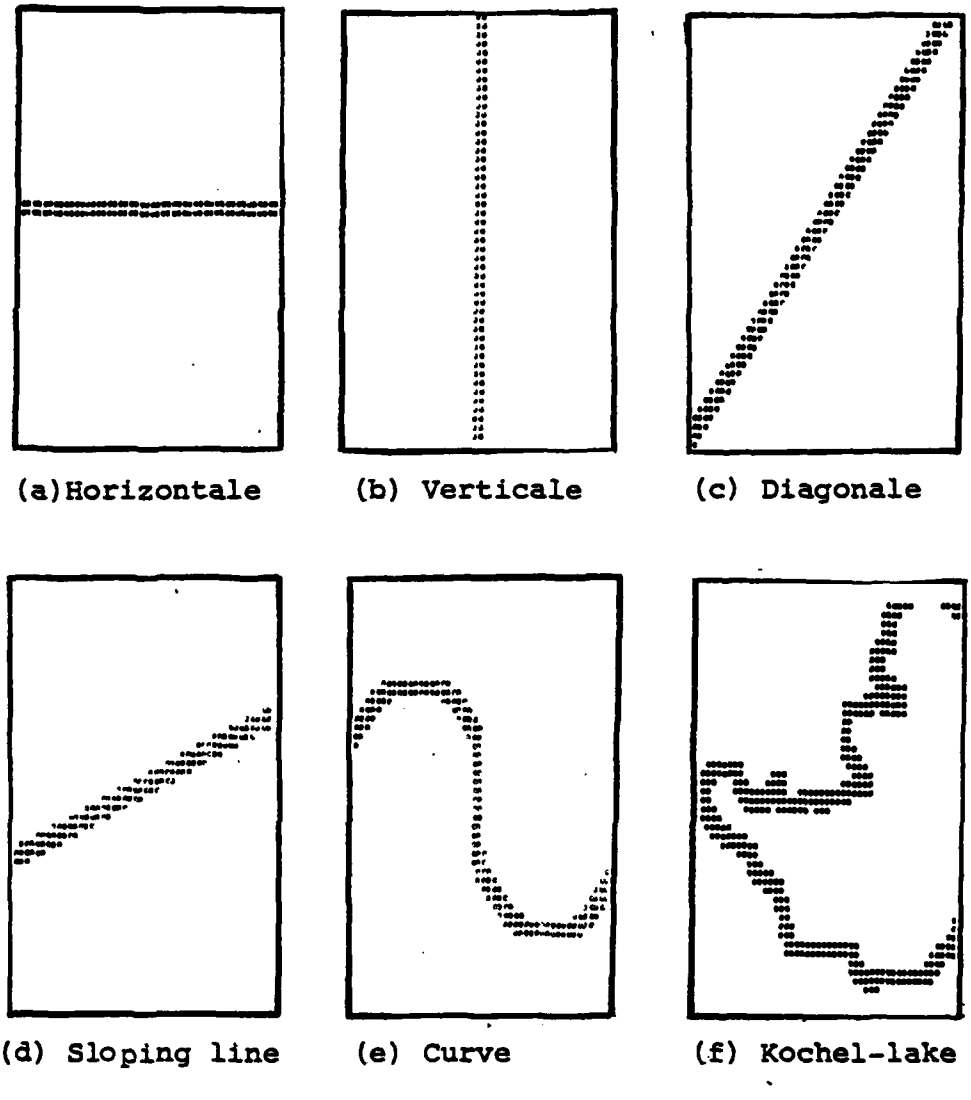
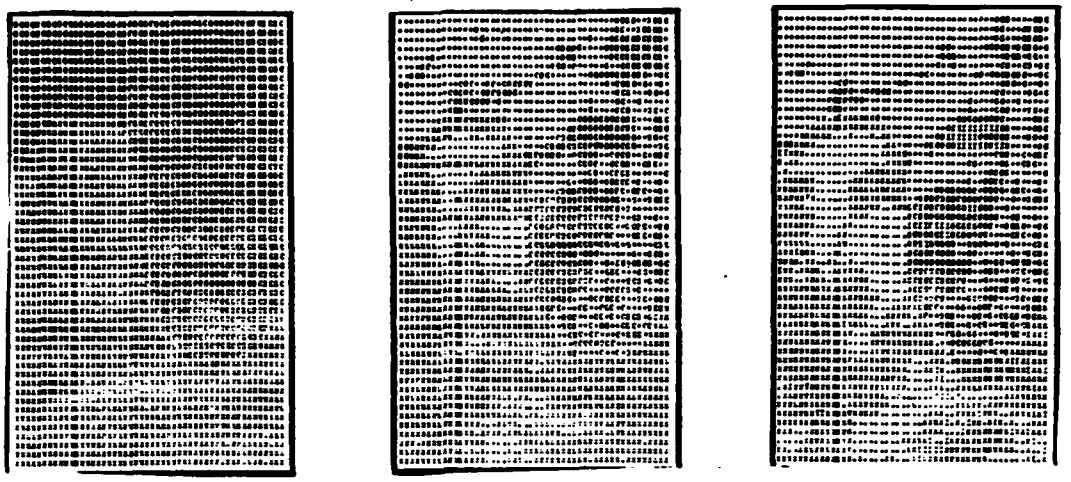
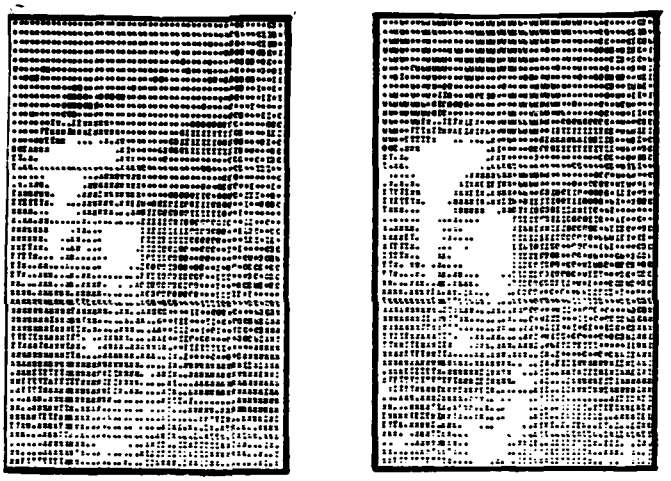


Figure 5.8: Edge reference images for the synthetic data of figure 5.6.



(a) clean f=ø (b) f=1 (c) f=2



(d) f=3 (e) f=4

Figure 5.9: Noise as used with synthetic image (curve).

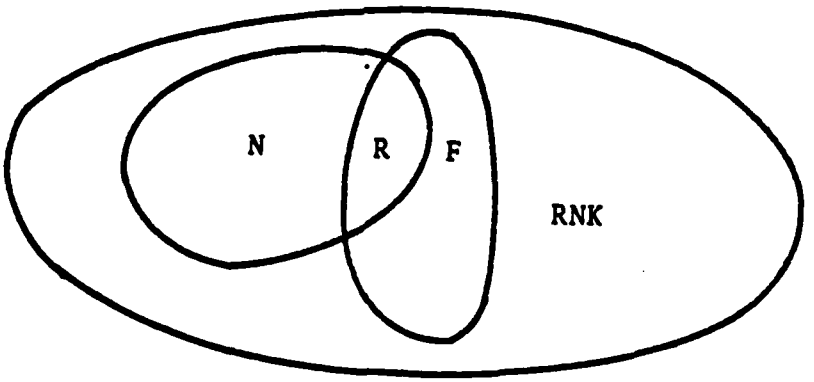


Figure 5.10: Types of sets of "edges" after edge detection.

(b) comparing position and number of edges in binary image with those in the reference image.

From this one has R correctly identified edges, N non-identified but existing edges with value $\neq 0$, and F incorrectly identified edges = $\emptyset$. Figure 5.10 illustrates the fact that the sets R,N,F are not disjunct.There are totally K edges that exist, NK non-edges and B image points.

We see that

$$\text{card (K)} = \text{card (N)} + \text{card (R)}$$
$$\text{card (NK)} = \text{card (RNK)} + \text{card (F)}$$
$$\text{card (B)} = \text{card (RNK)} + \text{card (N)} + \text{card (R)} + \text{card (F)}.$$

From this we compute

$$PR = 100 * \frac{\text{card (R)}}{\text{card (K)}} \; ; \quad PRNK = 100 * \frac{\text{card (RNK)}}{\text{card (NK)}}$$

and finally:

$$M = \frac{PR * \text{card (NK)} + PRNK * \text{card (K)}}{\text{card (NK)} + \text{card (K)}} \qquad (5.22)$$

Figure 5.11 shows the M-values for one test-image (f) and one operator (ROBERTS).

STOLR	60	80	100	120	140
0	100.00	100.00	46.56	46.56	7.30
1	100.00	90.18	48.74	35.65	10.57
2	96.91	84.18	52.55	36.20	14.39
3	99.50	86.26	56.91	34.56	20.39
4	98.86	90.82	60.07	39.43	24.74
MY	99.65	90.29	52.97	38.48	15.48

Figure 5.11: Means M of performance of edge operator of Roberts used on synthetic image Kochel.

Figure 5.12 shows an overview of all operators and all test-images, using the best threshold. Fig. 5.13 compares clean an noisy images.

BILD	KANTENOPERATOREN			
	ROSENFELD	ROBERTS	LAPLACE	HOR DIF.
HORIZONTALE	99.92	99.50	19.47	12.78
FLACHE	99.98	99.57	57.78	30.84
DIAGONALE	99.99	98.78	93.67	52.94
VERTIKALE	99.93	99.90	12.52	99.53
KURVE	99.83	99.14	71.72	53.02
KOCHELSEE	99.63	99.65	69.31	50.25
MITTELWERT	99.98	99.42	54.08	49.89

Figure 5.12: Maxima of mean M of performance factor.

BILD	KANTENOPERATOREN			
	ROSENFELD	ROBERTS	LAPLACE	HOR.DIF.
HORIZONTALE	100.00 (99.74)	100.00 (99.69)	1.96 (40.59)	1.96 (28.35)
FLACHE	100.00 (99.91)	100.00 (99.58)	51.31 (67.81)	26.97 (37.40)
DIAGONALE	100.00 (99.96)	97.93 (99.38)	100.00 (85.16)	51.41 (58.31)
VERTIKALE	100.00 (99.74)	100.00 (99.68)	1.96 (24.24)	100.00 (97.85)
KURVE	100.00 (99.84)	100.00 (99.43)	57.81 (88.48)	46.18 (64.94)
KOCHELSEE	99.96 (99.71)	100.00 (98.86)	59.45 (77.48)	45.47 (59.03)

Figure 5.13: Maxima of mean M in clean and noisy image (in brackets), using maximum noise factor of 4.

From all these results we can conclude

- thresholds have great importance;
- operators of Rosenfeld and Robers are superior;
- the simpler operators depend strongly on the direction of structures in the image.

We also found that with increasing noise the simpler algorithms improve: this is caused by giving correctly identified edges a higher weight in equation (5.22) than wrongly identified ones. With noise more correct edges are found, but also more incorrect ones. The better edge operators identify all edges with and without noise, but with noise they also produce wrong edges.

Computing times for the 4 investigated edge operators compare as follows:

Rosenfeld	Roberts	Laplace	horizontal difference
3.75	2.27	2.15	1

Since ROSENFELD and ROBERTS are practically equivalent in their overall performance it can be concluded that the method of ROBERTS is the most appropriate due to more modest computing requirements.

5.5 OTHER PRE-PROCESSING FUNTIONS

Numerous other techniques must be used for pre-processing prior to pattern analysis. The standard one are being used such as

- contrast manipulation;
- principal component transformation;
- ratioing of 2 images;
- generation of vector length images;
- clustering technique using principal component transform.

In addition a specific technique was developed to enhance features (areas) in a specific section of the image.

From external sources the approximate size and location of a feature is known. In the identified image area a histogram is produced in each image channel. Since the information on the feature is approximate only, the histogram will contain pixels not belonging to the feature.

To suppress the effect of those pixels that do not belong to the feature, one now selects a portion [A,B] of the histogram acc. to figure 5.14 so that a certain percentage of the pixels is within [A,B].

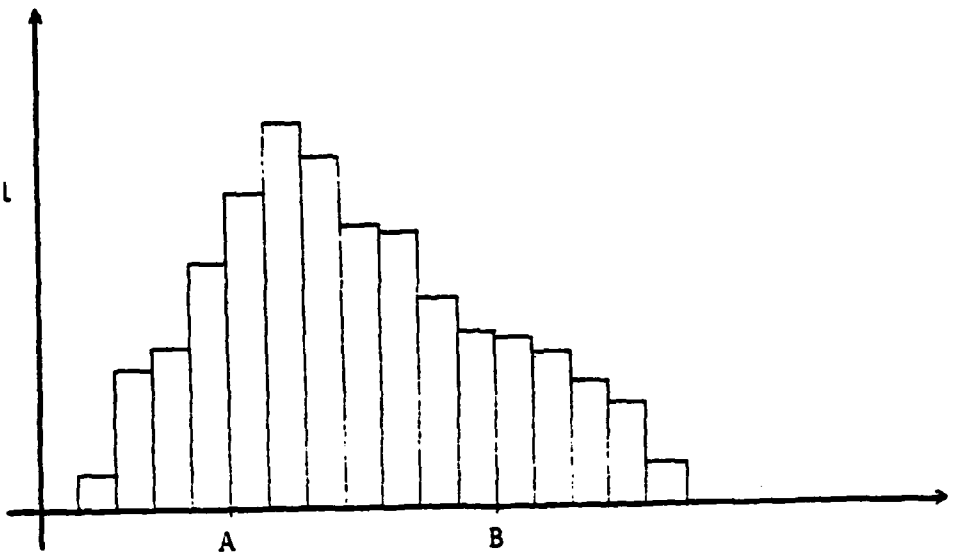


Figure 5.14: Range [A,B] contains 80 % of all values of the histogram.

Let us now denote with UG_k, k = 1, .. n the lower boundary of gray values (a) and OG_k, k = 1, ... n the upper one. k is the index for the channel.

We set

$$MINP = \prod_{k=1}^{n} UG_k \tag{5.23}$$

$$MAXP = \prod_{k=1}^{n} OG_k \tag{5.24}$$

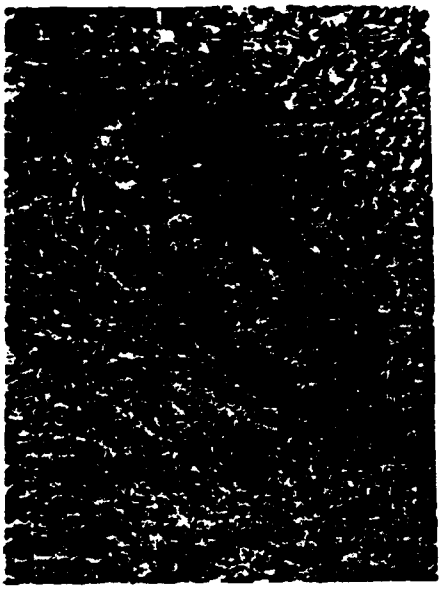
and obtain the transformation

$$T_{ij} = \left(\prod_{k=1}^{n} B_{ij}^{(k)} - MINP \right) * \frac{255}{MAXP-MINP} \tag{5.25}$$

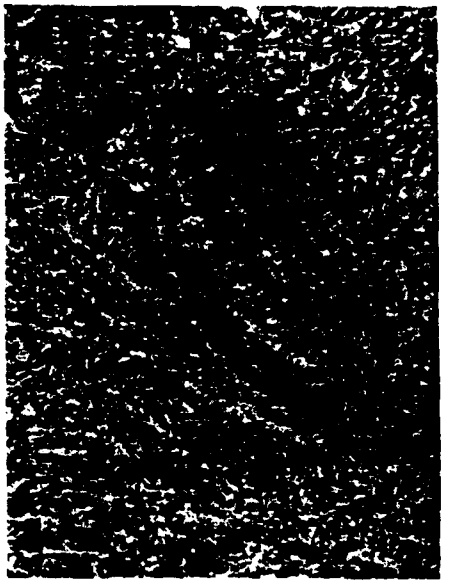
for all image points $B_{ij}^{(k)}$.

channel 4

channel 5



channel 6



channel 7

Figure 5.15(a)-(d): Four channels of a section of an original LANDSAT-image.

This transformation causes the image points with gray values similar to those of the feature used for computing MINP, MAXP to fall in the gray value range (0,255). All other points will be outside that range (either 0 or 255). This procedure works well with large areas such as lakes,forests, etc.. With linear features (roads) one has to transform with another, large object and then consider the 0 or 255 gray values.

Figure 5.15 shows 4 original LANDSAT-channels, 5.16 a vector length image, 5.17 the two first principal components PC1, PC2 and 5.18 a compressed image acc. to equation (5.25). For this river Mur was used to compute equations (5.23), (5.24) and this causes the free-way to fall into gray value 255.

Figure 5.16: Vector length image of scene in figure 5.15.

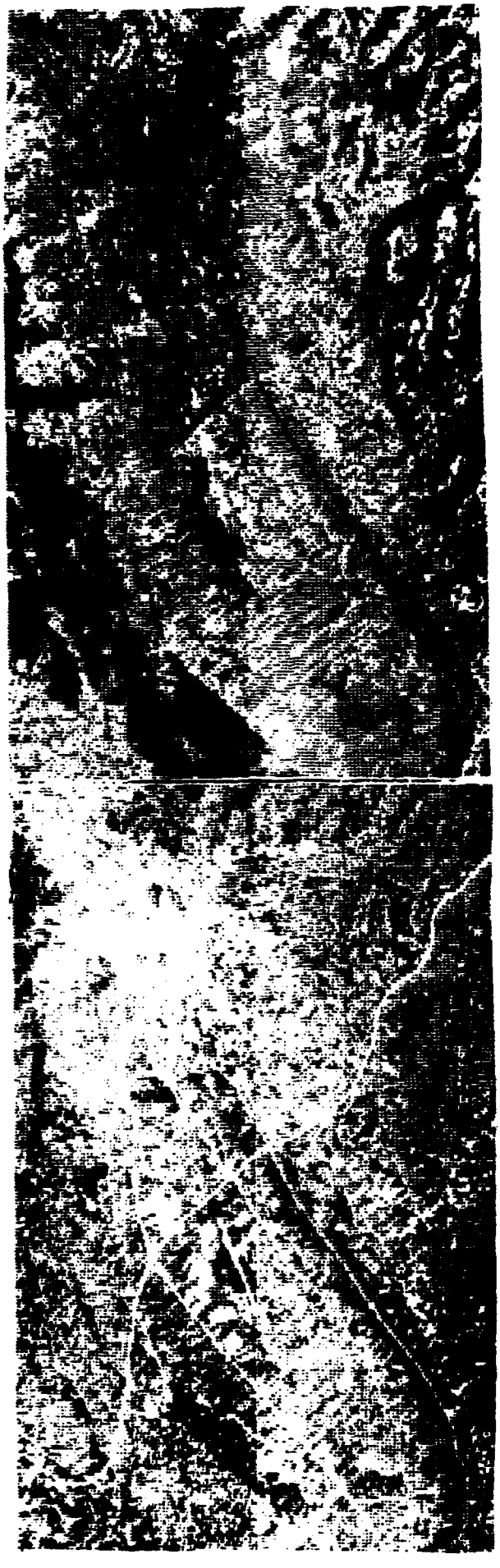


Figure 5.17: Two first principal components PC1(a), PC2(b) of scene of figure 5.15.

Figure 5.18: Compressed image using equ. (5.25) for water and the scene of figure 5.15.

6. RECOGNITON PROCEDURES

Recognition in our special case is concerned only with separate objects. These are objects which appear in a map and have either a uniform greyscale or a uniform texture. Images with objects of uniform texture can be transformed into images of uniform greyscale by one of the methods e.g. described by WESZKA (1979) where the resulting grey values are texture measures of a local environment. So we can concentrate on features in images of uniform grey-scale.

A second restriction to general recognition procedures comes from the fact that we have already a lot of information from the map concerning the object: we know the approximate location of the object, we know its shape, possible the range of grey values. It is therefore meaningful to start with a binary matrix of the shape of the object to use it as a mask to compare the appropriate grey values with the predicted value from the data bank or from other calculations on the image. The result of the comparison is then another binary matrix called " pattern ". These parameters of the first step of the recognition process are also shown in figure 2.7.

In a second step one has to decide whether the pattern found in step 1 represents the object or not. Here it is called VERIFICATION. It computes a measure of similarity between object and pattern. An object is denoted as rejected (not recognized) when the verification measure is below some accept treshold value.

Four methods for centering the map feature correctly over the image have been studied (see figure 2.7). Their principles of operation and implementations are described in the following chapters, together with the verification functions employed. These chapters are preceded by definitions of correlation functions, which are used in several of the procedures. Corresponding practical experiences, results and tests are presented in a separate chapter 8 so that the current chapter is merely theoretical.

6.1 CORRELATION FUNCTIONS WITH BINARY MATRICES

From the registration of an image with another image we know many correlation functions (DOWMAN, 1977;, BARNEA et al., 1972; ANUTA, 1970). Correlation functions with binary matrices are treated here because the formulas restrict themselves to expressions which use only standard statistical measures and masked image areas. This advantage has two effects: the number of accesses to the image elements can be minimized and standard procedures for statistical measures can be used.

Our formulas are derived from two different functions: The coefficient of correlation (KREYSZIG, 1973) and the cross-correlation (BARNEA et al., 1972).

The following statistical standard measures are used in our formulas (binary matrix and image S should have the same window size):

(6.1) $F(M) = \sum_{i,j} M(i,j)$ ..... surface of binary matrix M

(6.2) $F(S) = n * m$ ..... surface of an image window S of n columns and m rows

(6.3) $\mu(S) = (\sum_{i,j} S(i,j))/F(S)$.... mean value of image window S

(6.4) $\mu(M) = F(M)/F(S)$ ... mean value of binary matrix

(6.5) $\mu_M(S) = \left[\sum_{i,j}(S(i,j)*M(i,j))\right] /F(M)$ ... mean of image S over the binary mask M

(6.6) $\sigma^2(S) = \left[\sum_{i,j}(S(i,j) - \mu(S))^2\right] /(F(S)-1)$ ... variance of image S

6.1.1 Derivation of the coefficient of correlation

$$coco = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} (S(i,j) - \mu(S)) * (M(i,j) - \mu(M))}{\sqrt{\left[\sum_{i=1}^{n}\sum_{j=1}^{m}(S(i,j) - \mu(S))^2\right] * \left[\sum_{i=1}^{n}\sum_{j=1}^{m}(M(i,j) - \mu(M))^2\right]}} .$$

Building the product and splitting the sums simplifies the numerator rN:

$$rN = \sum_{i=1}^{n}\sum_{j=1}^{m} S(i,j)M(i,j) - \mu(S)\sum_{i=1}^{n}\sum_{j=1}^{m} M(i,j) - \mu(M)\sum_{i=1}^{n}\sum_{j=1}^{m} S(i,j) + \mu(S)\mu(M)F(S) .$$

Using equations (6.1), (6.3), (6.4) and (6.5) from above rN becomes

$$rN = F(M)\cdot\mu_M(S) - \mu(S)F(M) - F(M)/F(S) * F(S)\mu(S) + \mu(S)\frac{F(M)}{F(S)}F(S) .$$

It follows directly

$$rN = F(M)\mu_M(S) - F(M)\mu(S) - F(M)\mu(S) + F(M)\mu(S)$$

and

$$rN = F(M)(\mu_M(S) - \mu(S)).$$

With equ. (6.6) and knowing that $M^2(i,j)=M(i,j)$ the square rD^2 of the denominator rD becomes

$$rD^2 = (F(S)-1) * \sigma^2(S) * \left[\sum_{i,j} M(i,j) - F(S)\, \mu^2(M) \right]$$

and from (6.1) and (6.4) it derives

$$rD^2 = \left[F(S)-1\right] * \sigma^2(S) * \left[F(M) - F(S) * F^2(M)/F^2(S)\right]$$

and

$$rD^2 = \left[F(S)-1\right] * \sigma^2(S) * F(M)/F(S) * \left[F(S)-F(M)\right] .$$

Hence the quotient of rN and rD results to

$$(6.7)\quad coco = \frac{\mu_M(S) - \mu(S)}{\sigma(S)} \cdot \sqrt{\frac{F(M)\ F(S)}{\left[F(S)-1\right] * \left[F(S)-F(M)\right]}}$$

In an analogous way the formula for the cross-correlation derives from

$$cross = \frac{\sum_{i,j} M(i,j) * S(i,j)}{\sqrt{\left[\sum_{i,j} M^2(i,j)\right] * \left[\sum_{i,j} S^2(i,j)\right]}}$$

and results in

$$(6.8)\qquad cross = \mu_M(S) * \sqrt{\frac{F(M)}{F(S)\,\mu^2(S) + \left[F(S)-1\right]\sigma^2(S)}} .$$

6.1.2 Correlation between two binary matrices M_1, M_2

In the case of two binary matrices M_1 and M_2 only surfaces F(M) and F(S) are of interest. M is one of the two matrices or their product $M_1 . M_2$, and F(S) represents the minimal window size which comprises M_1 and M_2. Pixels outside of the window of one of the binary matrices are per definition set to Ø. Since the derivation of the formulas is trivial only the results are presented here.

$$(6.9) \qquad \text{bicoco} = \frac{F(S) * F(M_1 . M_2) - F(M_1) * F(M_2)}{\sqrt{F(M_1) * F(M_2) \left[F(S) - F(M_1)\right] * \left[F(S) - F(M_2)\right]}}$$

Binary cross-correlation:

$$(6.10) \qquad \text{bicross} = \frac{F(M_1 \cdot M_2)}{\sqrt{F(M_1) * F(M_2)}}$$

For comparison purposes formulas (6.7) and (6.9) of the coefficient of correlation are used as absolute values. All correlation measures which appear in the following chapters are given in percent, That means that the values are normalized between Ø and 1ØØ, where Ø stands for no correlation and 1ØØ for the best.

6.2 COMPARING THE REAL WITH A SYNTHETIC IMAGE

It is obvious that the binary image constructed from the map feature is a mask or a template or a synthetic image. This could be shifted over the window to find maximum correspondence between the window pixels and the template. This follows the sequential similarity detection algorithm of BARNEA and SILVERMAN (1972) which is normally applied to correlating two images. However, the method is typically suited to define a geometric distortion vector; it is not equally suitable to define the shape of a feature in the image.

One of the possible implementations of this shift algorithm is shown in figure 6.1.

It starts at the position of the binary matrix, which is estimated from the map. Correlation measures maxØ are calculated for levels of equal distance d from the original positic: It continues for ascending distances d until either no better correlation is found or a maximum distance Dmax is reached. The function binary matrix (pos) delivers the binary matrix shifted to position "pos". The condition in line 1Ø of figure 6.1 defines all positions of fixed distance d from the original "position" in the 8-neighbourhood (figure 6.2). The position "positmax" of the best match is used to shift the binary matrix to the resulting pattern.

```
(1)  procedure SHIFT (image, binary matrix, pattern);
(2)  begin
(3)     maxØ :=Ø;
(4)     max1 := 1ØØ * Abs(coco  (image , binary matrix));
(5)     d:= 1;
(6)     positmax:= position(binary matrix);
(7)     while max1 > maxØ or  d ≤ Dmax  do
(8)     begin
(9)        maxØ:= max1;
(10)       for all pos with (max(x(pos-position),y(pos-position))=d)do
(11)       begin
(12)          pattern:= binary to matrix (pos)
(13)          c:=1ØØ * Abs(coco  (image,pattern));
(14)          if c> max1 then
(15)          begin
(16)             max1:=c;
(17)             positmax:= pos;
(18)          end;
(19)       end;
(20)       d:= d+1;
(21)    end;
(22)    pattern:= binary matrix(positmax);
(23) end.
```

Figure 6.1: Recognition procedure SHIFT.

```
                - - - - -
                2 2 2 2 2
                2 1 1 1 2
        - - - - 2 1 Ø 1 2 - - - -
                2 1 1 1 2
                2 2 2 2 2
                - - - - -
```

Figure 6.2: Distances in the 8-neighbourhood.

6.3 REGION DETECTION BY THRESHOLDING

The principle of the SHIFT-method was to locate the real position of the object in the image by correlating the binary mask and the image at different horizontal and vertical positions. The shape of the object never changed, so that local distortions could not be detected. We tested another method of region recognition, which starts with a rudimentary binary feature and assembles pixels belonging to a region R according to the corresponding grey values in the picture.

The simplest method for assembling the pixels x_i belonging to a region R is by thresholding. Supposing that we start with the mask M from the map data bank projected on the image we calculate the following parameters before thresholding the image:

- a lower and an upper threshold value and
- a starting mask S containing all pixels which are possible candiates to belong to region R.

If we assume a dislocation of any pixel with a maximum of D pixel diameters the starting mask S can be computed from M by adding a band of D pixels around the feature. Then for all pixels of R, $S(i,j) = 1$ under that assumption (see fig.6.3b).

To choose the threshold values we tested various methods with different complexity. The simplest and fastest one is to take prefixed values. That is possible if the underlying image has been preprocessed so that all pixels belonging to a certain object type have a known range, which is then used as threshold interval [IPU, IPO] .

Sometimes we know only that the pixels of the object accumulate at one end of the grey scale. In this case a histogram can be used to find the other value. The histogram shall be taken from an image region which contains the whole object. This could be a rectangular region (S-window in figure 6.3a) D pixels greater in each axis direction than the original window of the object or the starting mask S as described above. Since the real surface of the object is known from our data bank the threshold value is defined by the condition that the number of grey values between the threshold pixels equals the surface content of the object. In this way procedure ADAHIS works in figure 6.3a. Factor p depends upon the dislocation D and the shape of the object. It is directly proportional to D and to the ratio $F(S)/F(M)$ where M stands for the binary matrix representing the object and S for the starting mask. $F(x)$ denotes the surface content of the respective parameter x.

If no information of the grey values of an object can be predicted, standard statistical measures produce sometimes acceptable results. We calculate the mean value MV

```
(1)  procedure THRESH (image, M,S)
(2)  begin
(3)     S-window:= Window(M)+(-D,-D,+2D,+2D);
(4)     STARTINGMASK (binmat,S-Window, D,S);
(5)     HISTO (image, S, S-Window);
(6)     ADAHIS (p*F(M), IPU, IPO);
(7)     for all (i,j)∈ S-Window do
(8)        if S(i,j)= 1&¬(IPU ≤image(i,j)≤IPO) then S(i,j):=Ø;
(9)  end.
```

Figure 6.3a: Recognition procedure THRESH.

```
(1)  procedure STARTINGMASK(binmat, window,D,mask)
(2)  begin
(3)     for all (i,j)∈ window do
(4)     begin
(5)        if binmat(i,j)=1 then mask (i,j):= 1 else
(6)        begin
(7)           d:=1;
(8)           sum:=Ø
(9)           while sum=Ø & d≤D do
(10)          begin
(11)             k:=d;
(12)             l:=Ø;
(13)             while k Ø & sum = Ø do
(14)             begin
(15)                sum:=sum+binmat(i+k,k+l)+
                              binmat(i-l,j+k)+
                              binmat(i-k,j-l)+
                              binmat (i+l,j-k);
(16)                k:=k-1;
(17)                l:=l+1;
(18)             end
(19)             d:=d+1;
(20)          end;
(21)          if sum=Ø then mask(i,j):=Ø else mask(i,j):=1;
(22)       end
(23)    end
(24) end.
```

Figure 6.3 b: Procedure STARTINGMASK

and the standard deviation SD of the grey values in a certain region. Then the threshold values are IPU:=MV-SD and IPO:=MV+SD respectively. The underlying region can be the projected object matrix M or the starting mask S or a region smaller than M or an image window containing M or only a portion o M.

After these preparations the proper thresholding is performed. Pixels from S are eliminated if the corresponding grey

value does not fall into the threshold interval (see lines 7 and 8 of figure 6.3a).

The disadvantage of the method THRESH is, that the resulting binary mask S is not necessarily a connected set. An advantage is the parallel operation of the algorithm.

6.4 ADAPTING THE HISTOGRAM

Another method which takes into account the disadvantage of thresholding starts with the contour of the object as given in the map. A histogram of the image over the binary region will reveal that pixels are included that should be eliminated. On the other hand, pixels in the remainder of the image are not part of the approximate binary projection, of the object, but are adjacent to it and should be added, because their grey values are very close to that of the feature. A search along the boundary of the feature will thus permit to connect the original contour, using the histogram to decide which pixels to reject and which ones to accept. This process can be repeated iteratively until either no corrections to the contour are done in the last step or a maximum iteration count is reached.

Figure 6.4 shows the procedure in more detail. MH is there a binary matrix which guarantees that changing the value

```
(1)  procedure ADAPT (image, M, binadapt, maxiter)
(2)  begin
(3)     S-Window:= Window(M)+(-maxiter,-maxiter, +2 maxiter, +2 maxiter)
(4)     iter:= 1;
(5)     alt:=-1; neu:= Ø;
(6)     binadapt:= M;
(7)     while iter ≤ maxiter & neu > alt  do
(8)     begin
(9)        alt:= neu;
(10)       MH:= (1) - binadapt;
(11)       HISTO (image, binadapt, S-Window);
(12)       ADAHIS (p, F(M), IPU, IPO);
(13)       for all pixel ∈ S-Window  do
(14)           if MH(pixel)=Ø v(pixel ∈ border (MH) & (IPU≤image(pixel)≤IPO))
                        then binadapt (pixel):= Ø
(15)                    else binadapt (pixel):= 1;
(16)       MH:=(1)- binadapt;
(17)       for all pixel ∈ S-Window  do
(18)           if MH (pixel)=Ø v(pixel ∈ border(MH)& not (IPU≤image (pixel)≤IPO))
                        then binadapt (pixel):= Ø
(19)                    else binadapt (pixel):= 1;
(20)       iter:= iter+1;
(21)       neu:= F(|binadapt - M|);
(22)    end
(23) end
```

Figure 6.4: Recognition procedure ADAPT.

of a pixel from 1 to Ø does not influence the property "pixel € border (MH)" during one iteration step. Lines (10) and (16) assign the binary complement of binadapt to MH. The selection of threshold values IPU and IPO in lines (11) and (12) works analogous to procedure THRESH. Condition "neu > alt" in line (7) and the calculation of "neu" in line (21) to be the surface content of the absolute binary difference between the original M and the adapted matrix "binadapt" express the termination of the loop if no more changes have accurred and "neu" is empty.

An advantage of this procedure lies in the fact that all changes of the binary matrix preserve its connectivity. The idea for this method derives from a skeletonizing or shrinking algorithm of ROSENFELD (1970). "ADAPT" combines the connectivity-preserving deletion process with an additional threshold condition on the image. ROSENFELD did show that the binary property of his algorithm preserves connectivity. Since we use only the deletable pixels from ROSENFELD his result must be a subset of ours. Hence our binary matrix has the same property.

A disadvantage of ADAPT is its relatively high processing time compared with thresholding. The algorithm has to inspect for each pixel a 3 x 3 pixel neighborship of the binary matrix to test for a border point (lines (14) and (18)) and in some cases the grey value in the image. This is done twice (for adding pixels and deleting pixels) with an additional histogram building at each iteration. The histogramming is not absolutely necessary at each iteration but it responds better to the fact that before adapting, a great amount of erroneous pixels affect the histogram and the choice of the threshold values IPU and IPO. In the version of figure 6.4 these are updated at each step in order to get the most homogeneous region in the image.

From these considerations follows that ADAPT should be used for the recognition of objects with small deviations from the expected location in the image to get a registration of greater precision than with other methods like threshold.

6.5 LINE DETECTION

Lines in satellite images appear either as borderlines between two distinct regions or as linear features like rivers, roads etc. Since borderlines are produced implicitly when their adjacent regions are recognized, their recognition is not of special interest for us. On the other hand they do not appear in our map data bank as separated objects. However, linear features do appear as skeletons in the map because they are considered to be very small regions.

Several procedures which detect lines in digital images are proposed in the literature. In principle there are 3 ways

to find lines from which the course is known approximately:

1) Shifting and rotating a binary mask over the image and determining the correct position by the maximum of a correlation function like method SHIFT in chapter 6.2. NACK (1975) and SWEDLOW (1978) correlated two binary edges images.
2) Thinning a binary region produced from the map projection of the skeleton followed by a thickening like the production of the starting mask in method THRESH. A thinning algorithm was proposed by ROSENFELD and DAVIS (1975). The thinning algorithm eliminates in our case the pixels according to their grey values like procedure ADAPT.
3) Following the lines sequentially.

The first two possibilities can be performed by procedures THRESH and ADAPT. We therefore concentrate in the following on sequential line following.

Literature (VANDERBRUG, 1973; DAVIS, 1973; HOLDERMANN, 1971; TENENBAUM, 1978) of sequential line following algorithms shows a great similarity with general problem solving algorithms in Artificial Intelligence (NILSSON, 1971; VEILLON, 1974). The algorithm in figure 6.5 represents a general line following algorithm as a result of adapting the general problem solving algorithm of VEILLON (1974) to the special problem of finding lines. All sequential line following algorithms that we found in the literature are expressable by this general procedure simply be selection of special parameters.

The algorithm LISU in figure 6.5 works on a general search space X. In our case this would be either a window of pixels containing the line or the set of pixels defined by the starting mask S as used in procedure THRESH (chapter 6.3). The search starts at pixel x_o in the search space X. This pixel must be determined approximately by the map projection of the line. Important parameters are the cost function c(x,y) defined on pairs x,y of neighbour pixels and the heuristic function h'(x). They essentially direct the search to a goal pixel $g \in X$ for which "end (g)" is true. The heuristic h'(x) estimates the sum of costs for the optimal path from the argument pixel x to the goal.

The state of the search is described by a set S of pixels from X and the approprate values φ and pointers "last". After each step, $\varphi(x)$ contains the minimal sum of costs for the optimal path from x_o to x. The path $(x_o, x_1 \ldots x_k, x)$ is defined recursively by "last":

$$\text{last}(x) = x_k \text{ and last}(x_i) = x_{i-1} \quad \text{for } i = 1, \ldots, k$$

and the set S contains all endpoints of minimal paths or in other words, all candidate pixels to continue to find the end

```
(1)  procedure LISU (x_o, X, end, c, h', x_n, last)
(2)  begin
(3)     x_n := x_o;
(4)     last (x_o) := nil;
(5)     for all x ∈ X  do φ(x) := ∞;
(6)     φ(x_o) := ∅;
(7)     S := { x_o } ;
(8)     while ¬ end (x_n) & S ≠ ∅  do
(9)     begin
(10)       eliminate x_n from S;
(11)       for all x ∈ ( Γ(x_n) - Γ(last(x_n)) )  do
(12)       begin
(13)          φ_n := φ(x_n) + c (x_n, x);
(14)          if φ_n < φ(x)  then
(15)          begin
(16)             φ(x) := φ_n;
(17)             enter x in S;
(18)             last (x) := x_n
(19)          end
(20)       end
(21)       m := ∞ ;
(22)       for all x ∈ S  do
(23)           if  φ(x) + h'(x) < m  then
(24)           begin
(25)              m := φ(x) + h'(x);
(26)              x_n := x
(27)           end
(28)    end
(29) end
```

Figure 6.5: A general line following algorithm.

pixel. These invariance conditions hold also for the goal state. Therefore the successfull result is defined by the pointer chain "last", starting at the goal pixel x_n and the minimal cost $\varphi(x_n)$ at that pixel. Condition S=∅ can be true only in the case of an error in the definition of X or if for no pixel x of X end (x) takes the value "true".

The importance of the heuristic lies in its tendency to prefer paths which tend towards the goal. Since $\varphi(x)$ contains the cost sum for the path from x_o to x and h'(x) estimates the cost sum for the rest of the path from x to a goal pixel, $\varphi(x)+h'(x)$ represents in line (23) an estimate for the path passing through x. The pixel x_n of S where the search shall continue at the next step is selected in lines (22) until (27) to be on the path for which minimal total costs are expected.

This informal description of the algorithm is based on the proof which can be found in VEILLON (1974).

6.5.1 Cost and heuristic function for some special methods

One method often mentioned in the literature is the "heuristic search" of a line. Cost and heuristic functions $c(x,y)$, $h'(x)$ are determined according to gray values $g(x)$ and $g(y)$. MARTELLI (see VANDERBRUG, 1973) uses as search space X the set of all possible edges on the image B. The set of neighbors on X is defined as follows:

If a,b,c,d,e and f denote points of image B in one of the following constellations,

```
c  d                    f a c
a  b        or          e b d
f  e
```

then the neighbors of edge (a,b) are enumerated in:

$$\Gamma((a,b)) := \{(a,c),(a,f),(b,d),(b,e),(c,d),(e,f)\} \quad .$$

The cost function is calculated

$$c((a,b),(x,y)) := \max\{\emptyset,\ 2M-(g(a)+g(x))+(g(b)+g(y))\}$$

with M to be the contrast $g(p)-g(q)$ of an optimal edge (p,q). The search starts with an edge $x_o = ((1,s_o),(1,s_o+1))$ behind column s_o in the first line of the picture and ends when

$$\text{end}((a,b),(x,y)) := (a=\text{last line})\&(x=\text{last line})$$

becomes true.

The algorithm uses further the following heuristic:

$$h'((a,b)) := \begin{cases} \emptyset & \text{if} \quad \text{line}(a) \geq \max\{\text{line}(x) \mid (x,y) \in S\} - 3 \\ \infty & \text{else} \end{cases}$$

The heuristic prevents the search to follow paths from end edges which lie 3 lines before the highest line number which had ever been developed and stored in set S during the preceding search. In effect backtracking is reduced and the search is forced towards the goal, the last line.

The Look-Ahead algorithms and the algorithm of CHERNIAVSKY (VANDERBRUG, 1973) use a similar cost function and estimate the heuristic $h'(x)$ in some neighbourhood $U(x)$.

An other efficient group of search methods are summarized under the notion "Directed Search". The search is there directed by external informations, which cannot be calculated directly on

the image and come therefore from the outside. They determine the heuristic h'(x) and influence via h' the flow of the search. A special form of the "Directed Search" is called "Planning" (KELLY, 1971; DAVIS, 1973). It assumes that intermediate goals are known from preceding steps and uses them through the heuristic to follow the line passing through.

HARLOW (in DAVIS, 1973) describes an algorithm, which gets its information directing the search from a description of the scene. This early approach could be regarded very near to ours in which a map knowledge shall help to find lines in satellite images.

A method proposed by MONTANARI (VANDERBRUG, 1973), which falls into the category of "Dynamic Programming", is in reality a simplification of the general algorithm by setting the heuristic h' equal to zero for all arguments. Further restrictions arise for the set of neighbors allowing only neighbors in the original sense for which certain conditions for their grey values are fulfilled. The search space is here built by the image pixels B itself.

A similarity between the "Bidirectional Search" (VANDERBRUG, 1973) and the scheme of the general sequential line following algorithm is not obvious. The search starts there simultaneously from start and end point of the line. For the starting point a set S of alternatives is created and for the end point a set T for paths running into the goal. A difficulty arises for the computation of the "end"-function which has the form:

$$\text{end}(x) \iff x \in S \wedge T$$

This derives from the fact that the function must be calculated at each step and requires an inspection of every element of S and T.

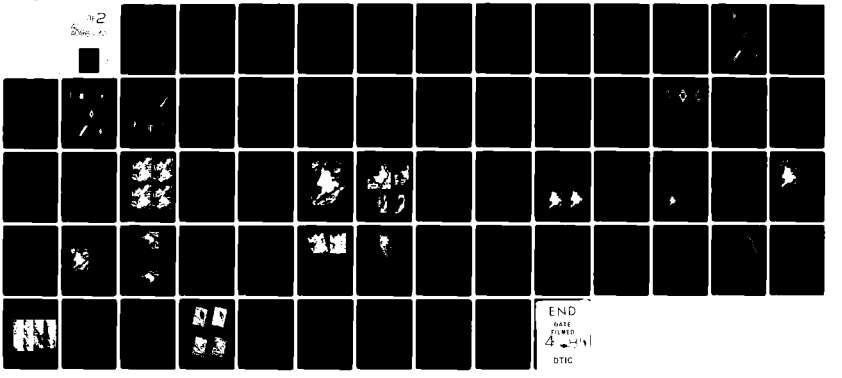
HOLDERMANN proposed in 1971 a straightforward method. His algorithm selects edges in the search space giving them two values: a weight and a direction. The neighbourhood $\Gamma(x)$ has the form of a sector centered in x of several pixel radius. Its main direction is the same as the direction of edge x. The characteristics of the cost function must be selected so that

1) the costs for a chosen alternative are always lower than for other alternatives,
2) a selected edge corresponds best to its predecessor in weight and direction and
3) if there is no direct successor, adjoining sectors are selected.

The heuristic has no importance in this approach.

AD-A096 132 TECHNISCHE UNIV GRAZ (AUSTRIA) INST FOR NATIONAL SUR--ETC F/G 9/2
AUTOMATIC REGISTRATION OF SCANNED SATELLITE IMAGERY WITH A DIGI--ETC(U)
NOV 80 F W LEBERL, W KROPATSCH DA-ERO-78-G-044

UNCLASSIFIED NL



END
DATE FILMED
4-81
DTIC

6.5.2 Interrelationship between cost and heuristic functions

VEILLON (1974) enumerates 2 conditions for costs c and heuristic h':

(A) $c(x,y) > \emptyset$ for all x,y of X and $x \in \Gamma(y)$

(B) $$\emptyset \leq h'(x) \leq \min \left\{ \sum_{i=1}^{n-1} c(x_i, x_{i+1}) \,\middle|\, x=x_1 \;\&\; \mathrm{end}(x_n) \;\&\; x_{i+1} \in \Gamma(x_i), i=1,n-1 \right\}$$

In the case of line detection the costs c depend essentially on the grey values of the corresponding pixels. If we assume that the image has been preprocessed so that pixels on the line have high grey values, then the desired path has the property that the sum of grey value differences to the maximum is minimal. This leads to the conclusion, that c(x,y) depends only on the grey value g(y). We further restricted the functional class of c to the form

$$c(x,y) = cg(g(y)) = (a + b * g(y))^n \text{ with } a,b,n \in R \text{ and } n > \emptyset.$$

The choice of c was a result of extensive tests and the consideration that pixels near the optimal grey value should yield small costs while grey values far away from the optimal one should be considered only in the worst case. Parameters a, b and n are determined by 3 conditions which can be derived from our data base knowledge if we assume that h'(x) measures the Euclidean distance from the goal.

1. If B is the highest grey value of the search space X then we set

   (B1) $cg(B + 1) = \emptyset$.

2. Since pixels of the line accumulate at B, we choose a range $[C,B]$ of grey values in which all line pixels should lie. Let H(g) denote the histogram of the grey values g in the search space X. Then

   $\sum_{g=C}^{B} H(g) = L$ can be used to determine C if L is the length of the line in units of pixels. To guarantee general condition (B), we demand that cg (x) for $x \in [C,B]$ should be 1 in the mean and therefore:

   (B2) $$\frac{\sum_{g=c}^{B} H(g) * cg(g)}{L} = 1 \quad .$$

3. Pixels outside of the line should get high costs. We set the mean costs for such pixels x with $g(x) \in [A, C-1]$

equal to the total sum L over all goal pixels.
"A" stands here for the minimal grey value. That means:

$$(B3) \quad \frac{\sum_{g=A}^{c-1} H(g) * cg(g)}{\sum_{g=A}^{c-1} H(g)} = L \ .$$

From these 3 condidtions parameters a,b and n are derived delivering a consistent pair of costs and heuristics (cg,h').

6.5.3 Representation of the result of Line following

The result of procedure LISU consists in a pointer chain "last" from a goal pixel x_n back to the starting point,. The transformation in a binary matrix "binline" is done in lines (8) to (13) of procedure LINDET (figure 6.6). Procedure COSTPARAMETER determines the values of a, b and n with the

```
 (1)  procedure LINDET (image, mapline, start, goal, binline);
 (2)  begin
 (3)     S-Window:= Window (mapline)+(-D,-D,+2D,+2D);
 (4)     STARTINGMASK (mapline, S-Window, D, Smask)
 (5)     HISTO (image, Smask, S-Window)
 (6)     COSTPARAMETER (a, b, n);
 (7)     LISU (start, Smask, end, c, h', x_n, last);
 (8)     for all x ∈ S-Window  do binline (x):=  ∅;
 (9)     while x_n ≠ nil  do
(10)     begin
(11)        binline (x_n):= 1;
(12)        x_n:= last (x_n)
(13)     end
(14)  end.
```

Figure 6.6: Filling a binary matrix "binline" with pixels of a line.

aid of the histogram built in line (5) according to conditions (B1), (B2) and (B3) (chapter 6.5.2). Function end (x) in procedure LISU (figure 6.5) is defined by

$$end(x) := (h'(x) = \emptyset) \ ,$$

and heuristic h' of a pixel x = (col, row) is computed as follows

$$h'(x) := \min \left\{ \sqrt{(col-gc)^2+(row-gr)^2} \;\middle|\; (gc,gr) \in goal \right\} .$$

The cost function c has the form as described in chapter 6.5.2.

6.5.4 Performance of sequential line detection

Sequential line detection has its origin in a general problem solving algorithm, which searches in a predefined search space of possible solutions for the best under a certain valuation criterion. How fast algorithm finds the result depends essentially on the choice of heuristic h' or as in our case - of the deduced cost function. If the perfect heuristic h were known, that is if condition (B) in chapter 6.5.2 takes equality on the right side, then the algorithm would find the optimal solution $(x_o, x_1, \ldots, x_n)$ exactly in n steps.

A disadvantage of these algorithms is the enormous computational effort if the line does not lie in the expected region. Then almost the whole search space must be inspected. It is therefore of great interest to keep the search space X as small as possible.

6.6 VERIFICATION OF FEATURES

Features or objects appearing in maps and images are determined by several characteristics (form, uniform grey value). In step 1 of the recognition process some of them were used to extract a feature (binary matrix) from the image. Verification is then the second step which tests if other characteristics describing the object have been violated in step 1 or not.

Since three of our recognition procedures (THRESH,ADAPT, LINDET) are adapting the form of the object to yield the best attainable uniform grey scale distribution, the resulting shape of the feature must be compared to the original one from the map. Figure 6.7 shows in lines (7) until (10) the general method of operation. Measures of the object and the pattern are computed in the appropriate coordinate system in order to construct a linear transformation function T which superimposes the transformed object over the pattern. Then the shape of the two features can be compared and measured by a correlation function.

Some of these measures, their computation and the reconstruction of a linear transformation will be described in

chapter 6.6.1. Correlation functions to measure similarity of the two features are presented in chapter 6.6.2.

```
(1)  procedure VERIFICATION (object, image, pattern);
(2)  begin
(3)     if search method (object)= 1  then
(4)     VERIFICATION:= coco(image, pattern)
(5)                                    else
(6)     begin
(7)        objectmeasures:= measures (object);
(8)        featuremeasures := measures (pattern);
(9)        BESTOVERLAY (objectmeasures, featuremeasures, T);
(10)       VERIFICATION:= correlation (T(object),pattern);
(11)    end;
(12) end.
```

Figure 6.7: Computation of a measure for verification.

The first recognition procedure SHIFT does not vary the shape of the feature. It is therefore meaningless to compare the two (in that case identical) shapes. We use either the correlation between grey values of the picture and the binary pattern that has already been calculated in procedure SHIFT or we correlate the pattern with another channel of the multiple (multispectral) image or a pre-processed version of it.

6.6.1 Some measures describing patterns, their computation and application.

(a) Translation and Center of Gravity

The measures or quantities discussed here have the purpose to establish a linear transformation

$$\underline{T}\,(\underline{x}) = \underline{A}\,\underline{x} + \underline{b}$$

from the map coordinate system to the image system. These measures are a derived quantity, obtained from possibly erroneous pixels that are defined to belong to a feature. Errors in the pixels should have a minimal effect on the derived measure.

The simplest measure we use is the <u>center of gravity</u> of a pattern. Its computation for a closed polygon P employes the formula

$$Sx(P) := \frac{1}{3}\left[\sum_{i=1}^{n}(x_i+x_{i+1})(x_{i+1}y_i - x_iy_{i+1})\right] / \sum_{i=1}^{n}(x_{i+1}y_i - x_iy_{i+1})$$

$$Sy(P) := \frac{1}{3}\left[\sum_{i=1}^{n}(y_i+y_{i+1})(x_{i+1}y_i - x_iy_{i+1})\right] / \sum_{i=1}^{n}(x_{i+1}y_i - x_iy_{i+1})$$

where (x_i, y_i), i = 1, ...n are the points of the polygon P, and (x_{n+1}, y_{n+1}) is identical to (x_1, y_1).

If a region R is described by two polygons P_1 and P_2, P_1 enclosed in P_2 (see figure 6.8), then the

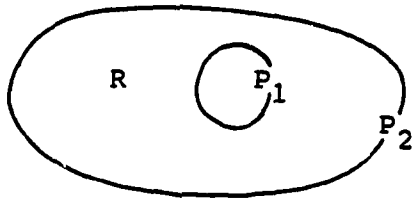


Figure 6.8: The shape of region R containing an island is described by the polygon P_1 and P_2.

center of gravity s(R) of R is computed following

$$\underline{S}(R) := \frac{\underline{S}(P_2)*F(P_2) - \underline{S}(P_1)*F(P_1)}{F(P_2) - F(P_1)}$$

with $F(P_1)$, $F(P_2)$ denote the areas of polygons P_1 and P_2, respectively.

The center of gravity of a binary matrix M computes as follows

$$\underline{S}(M) := \left(\sum_{\underline{x} \in Window(M)} \underline{x} * M(\underline{x}) \right) / F(M) \quad .$$

where F(M) is here the number of pixels in the pattern (M(x) = 1).

With the two centers of gravity, namely of region R in the map $\underline{S}(\underline{R})$, and of pattern M in the image, $\underline{S}(\underline{M})$, the shift vector $\underline{b}_o$ of the original transformation T_o

$$\underline{T}_o(\underline{x}) = \underline{A}_o \cdot \underline{x} + \underline{b}_o$$

can be corrected to be

$$\underline{b}_N := \underline{S}(M) - \underline{A}_o \cdot \underline{S}(R) \quad .$$

The new transformation T_N can be formulated as follows

$$\underline{T}_N(\underline{x}) := \underline{A}_o \cdot (\underline{x} - \underline{S}(R)) + \underline{S}(M) \quad .$$

(b) Rotation and scale with lines.

Other measures can be calculated from lines. Two reference points $\underline{P}$, $\underline{Q}$ in image coordinates and $\underline{P}'$ and $\underline{Q}'$ in map coordinates are used to correct the original transformation

$$\underline{T}_O(\underline{x}) = \underline{A}_O \cdot \underline{x} + \underline{b}_O$$

to

$$\underline{T}_N(\underline{x}) = \underline{A}_N \cdot \underline{x} + \underline{b}_N$$

with

$$\underline{A}_N = s \cdot \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \cdot \underline{A}_O = \underline{D} \cdot \underline{A}_O \quad .$$

The correction matrice $\underline{D}$ can be calculated from

$$\underline{Q} - \underline{P} = \underline{T}_N(\underline{Q}) - \underline{T}_N(\underline{P}') = \underline{D} \cdot (\underline{T}_O(\underline{Q}') - \underline{T}_O(\underline{P}')) .$$

Having the new $\underline{A}_N$, $\underline{b}_N$ computes as for translation:

$$\underline{b}_N = \underline{P} - \underline{A}_N \cdot \underline{P}'$$

6.6.2 Measures of correlation between object and pattern

The measures mentioned here are used to determine how well the shape of the (best-)transformed object corresponds to the shape of the pattern in the image. We assume that the object is transformed linearly so that the registration of object and pattern is optimal.

Then the polygonal representation is transformed to the raster of the pattern. A measure can be obtained using the binary coefficient of correlation from equ.(6.9), "bicoco", or the binary cross-correlation from equ. (6.10), "bicross", of the two binary matrices (see chapter 6.1).

Another method we used earlier was based on the number "differ" of pixels which do not appear in both matrices M_1 and M_2:

$$\text{differ} := \frac{F(|M_1 - M_2|)}{F(M_1)} \quad .$$

Tests showed that this approach is very sensitive to the total surface of M_2. The best results were obtained from the cross-correlation "bicross" which is simple to calculate and less dependent from the total area of a pattern or object.

7. CONTROL POINTS AND RECTIFICATION

The original problem of registering a map with the corresponding satellite image was partitioned in our approach to search first for registration of separate identifiable objects. From each such object we extract one or more pairs of corresponding points in map and image, which we call control points. During the process of recognizing objects in the image these control points are collected. If the set of control points covers the overlay region with sufficient density and an even distribution, it is used to geometrically transform the image to the map coordinate system.

Procedure CONTROLP (fig. 7.1) summarizes the selection of control points from an identified object and their collection in a file. Regions and lines are handled separately because of the difference in structure. From a region we extract one controlpoint, the center of gravity. The degenerated case of a circular region demonstrates why we restrict ourselves to only one point. From lines we extract at least two points.

```
(1)  procedure CONTROLP (object, pattern, controlpoints)
(2)  begin
(3)     if search method (object)= 4 then
(4)     begin
           { Lines };
(5)        controlpoints:=controlpoints+(I1(object),I1(pattern))
                                        +(I2(object),I2(pattern));
(6)     end                             else
(7)     begin
           { regions };
(8)        controlpoints:=controlpoints+(S(object),S(pattern));
(9)     end
(10) end.
```

Figure 7.1: Definition of control points

We could use the endpoints, but they are affected with errors because lines are often segments of longer lines for which the starting point in the image is not exactly defined. To eliminate this disadvantage and source of errors the line is subdivided into three segments of equal length. The intersection points I_1, I_2 of adjacent segments are then used as control points.

The registration process REGISTER is followed by a geometric transformation RECTIFY (figure 7.2) of the image into the coordinate system of the map. The transformation is done in a standard procedure with the aid of the control points. These are used to define a regular grid of anchor points in the rectified and in the distorted image. The grid is stored as a matrix of point coordinates in the original image. Rectification is done by bi-linear resampling within each grid mesh as defined by the anchor points. While the geometric location of the center of a new

```
(1)  procedure RECTIFY ( old image, controlpoints, new image)
(2)  begin
(3)     determine rectangular transformation grid (controlpoints);
(4)     for all rectangles of grid  do
(5)     begin
(6)        for all pixel of rectangle  do
(7)        begin
(8)           interpolate position(pixel) linearly in old image;
(9)           new image(pixel):= oldimage (position(pixel));
(10)       end
(11)    end
(12) end
```

Figure 7.2: Geometrical rectification of an image

pixel in the distorted image is computed by bi-linear transformation, its grey value is assigned as that of the next neighbour. The entire rectification is a well-documented standard routine (e.g. KONECNY, 1975) and was programmed in our case by WIESEL (1977).

8. RESULTS AND EXPERIENCES

8.1 INTRODUCTION

Numerical results and experiences were obtained using 3 different images:

(a) A synthetic image (denoted MERK).

(b) A section of a multispectral Landsat image of an area in Southern Germany with lakes (denoted WALCH).

(c) A section of a Landsat image in Southern Austria of an urban area (denoted GRAZ).

Tests and results will be described in this chapter; they will demonstrate in some detail the basic concept of ARSIM. In each of the three test images several specific aspects of the ARSIM concept have been evaluated.

Experiments with the synthetic image MERK serve to prove the basic validity of the procedures, using a well controlled ideal data set.

Test area WALCH was used originally for understanding the problem of feature recognition in satellite images. It served to develop the rudimentary algorithms and to select methods which promised some success.

Test area GRAZ finally had the function of verifying the developed methods with real data. This is the material to permit identification of current limits and needed improvements in the experimental system. The fundamental experiences can be summarized as follows:

- ARSIM is a valid concept to automatically merge a map with an image.

- The task of automatically merging maps and satellite images is a formidable one. There is no limit to the number of possible pattern recognition techniques that can be used.

- Successful feature recognition depends on specific preprocessing to prepare the feature for a recognition algorithm. Preprocessing must be turned to both the image and to the object to be recognized.

- Successful feature recognition also depends on a map data base of an appropriate organisation and content such that a relation to images can be established.

- The success of the approach depends on the implementation of experiences into the selection of preprocessing and recognition algorithms. A "learning system" is required.

- ARSIM must be organized in a highly modular form so that many different algorithms can be implemented as new experiences are gained.

8.2 EXPERIENCES WITH THE SYNTHETIC IMAGE

The purpose of working with a synthetic image and related data base is to have control over perturbation factors. Therefore the results can be more easily analyzed and the limitations of the concept can be understood.

8.2.1 The synthetic image MERK

Figure 8.1 shows in the form of a printer output the configuration of the five objects RECHT, DREI, KARO, PARA, KREIS in the image. Different geometric forms of objects have been used: RECHT is a rectangle, DREI a triangle, KARO a quadrangle with a rectangular hole, PARA a long parallelogram and KREIS is a circle. Pixels of the objects were set to grey value Ø (black) and the background to 255 (white).

A map data base was created showing the same features with identical geometry. Figures 8.2 and 8.3 illustrate the map.

Project funds did not permit us to fully exploit the possibilities of the synthetic images. Instead of a wide variety of perturbed versions, we could only test the influence of grey level noise generating a noisy picture as shown in figure 8.4.

8.2.2 A projection of the map onto the image

To simulate the geometric conditions from an approximately rectified satellite image the map data were geometrically deformed using the following linear transformation (referenced as projection (8.1) in the following chapters):

$$T_x(x,y) = -Ø.Ø35 * x + 1.Ø33 * y + 3.5ØØ$$
$$T_y(x,y) = 1.Ø58 * x + Ø.Ø4Ø * y - 5.ØØØ \qquad (8.1)$$

Figure 8.5 shows the projection of the five objects into the image. Table 8.1 enumerates the objects with their surface in the map and after transformation (8.1) including a discretization error and the deviation of the center of gravity from the ideal. These deviations are measured with the Euclidean distance. The assumption is met of a maximum shift of 1Ø pixels in each coordinate direction, except for object DREI which is used to study a more extreme case of geometric deformation.

Feature	Surface orig.	Surface project.	Differences in position(pixels)
RECHT	36.1	41.9	5.Ø99
DREI	2Ø.Ø	24.6	11.7ØØ
KARO	48.4	44.6	7.211
PARA	74.1	87.5	4.123
KREIS	48.5	55.8	12.Ø42

Table 8.1: Characteristics of the projection in pixel units.

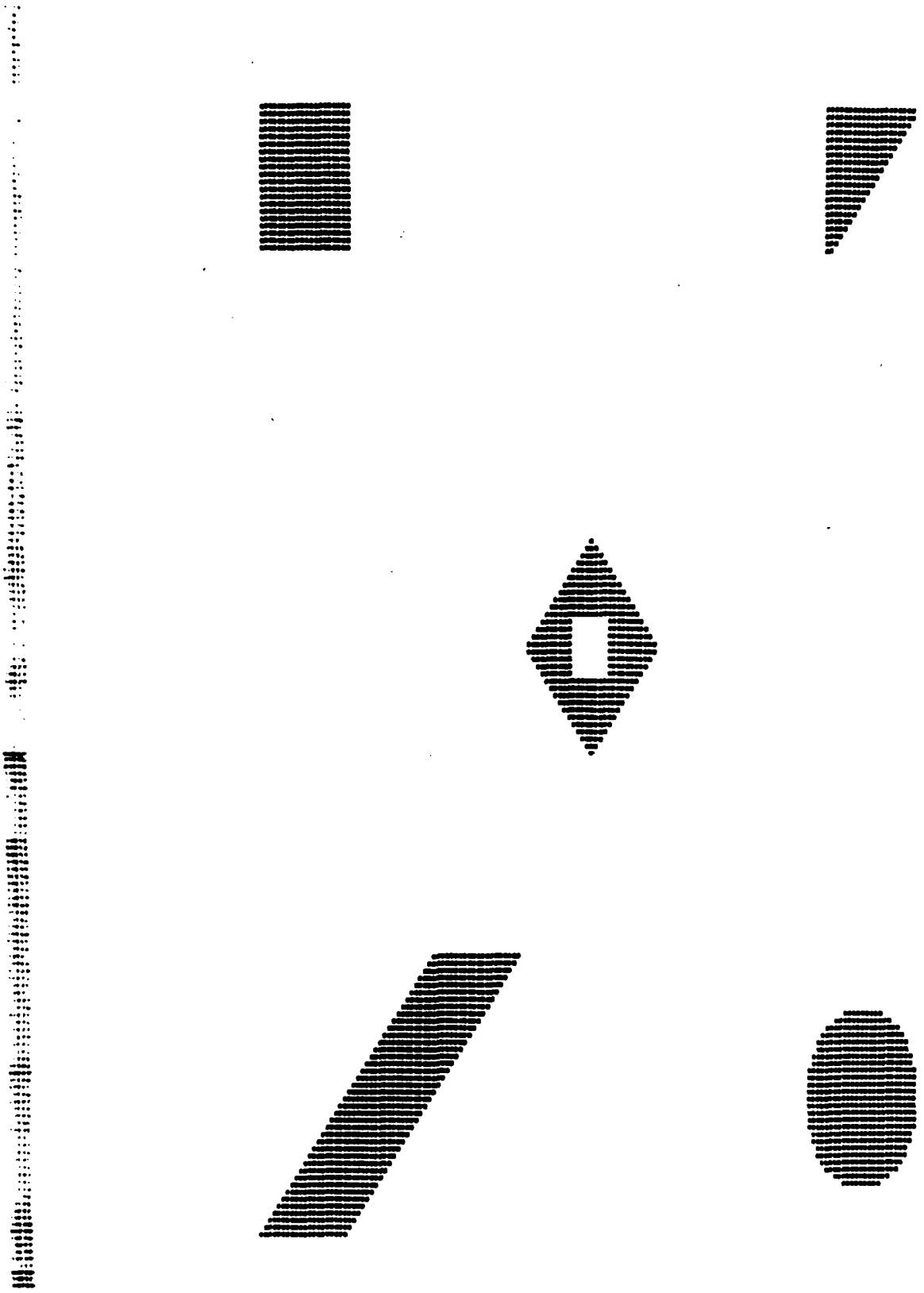


Figure 8.1: Synthetic image MERK.

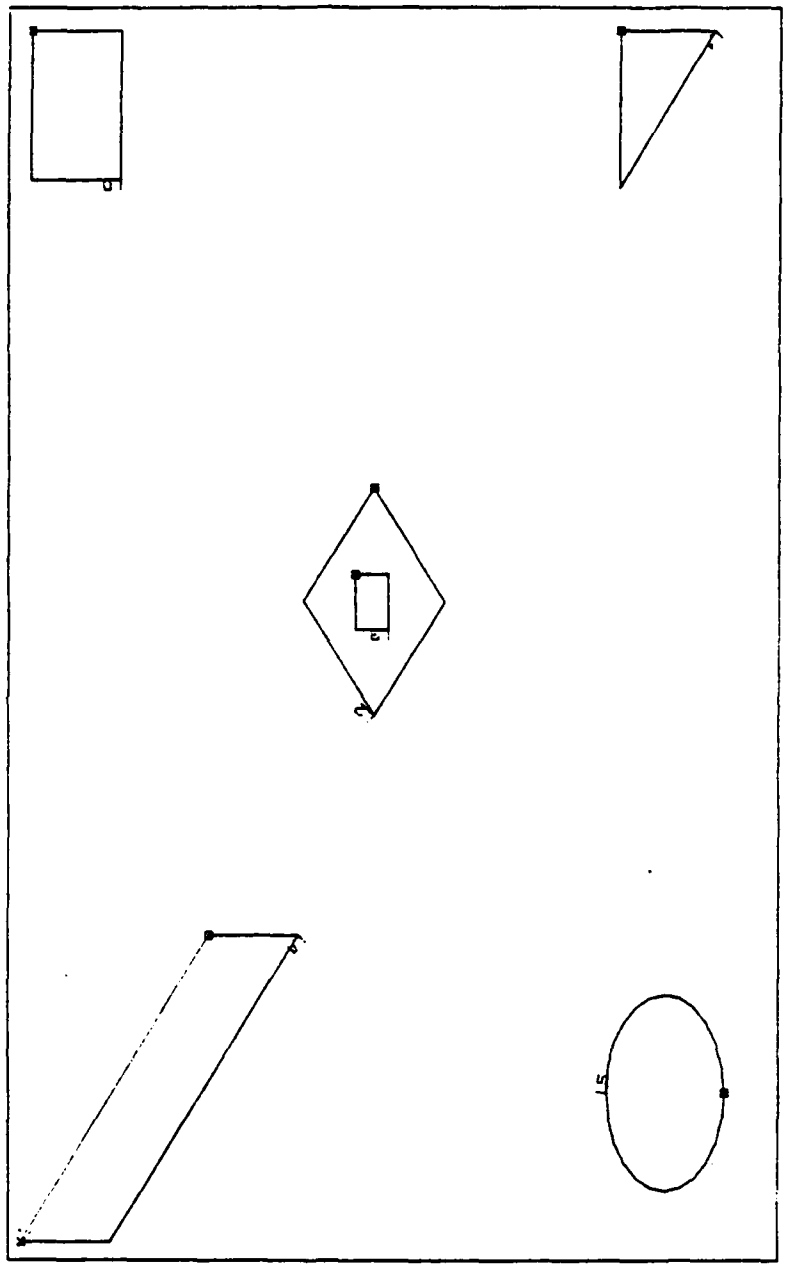
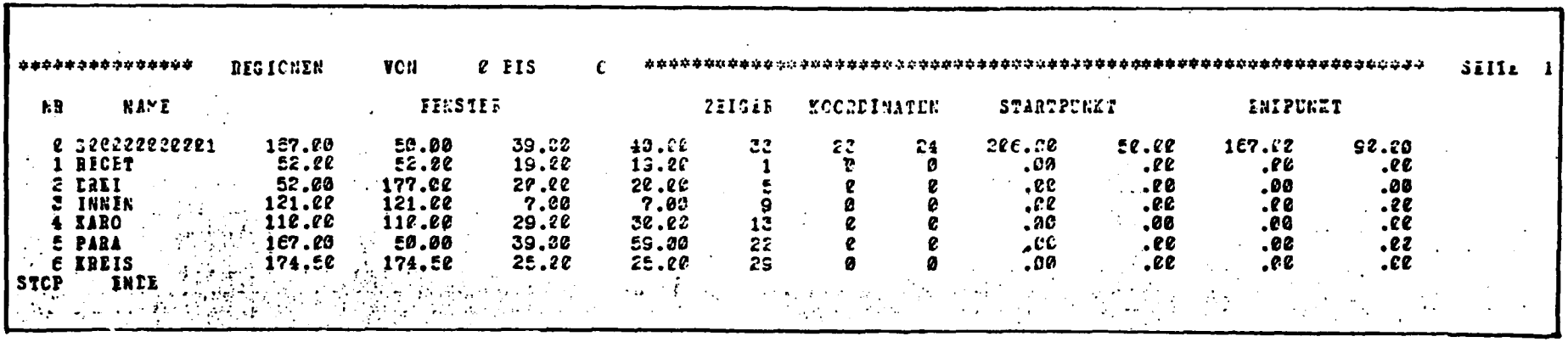


Figure 8.2: Plot of map MERK.



```
**********  REGIONEN   VON   0 BIS   6  **********                    SEITE  1

 NR     NAME                FENSTER                ZEIGER  KOORDINATEN   STARTPUNKT        ENDPUNKT

  0 3202222020201    167.00   50.00   39.00   40.00   33   22   24  206.00   50.00  167.00   92.00
  1 RECHT             52.00   52.00   19.00   13.00    1    0    0     .00     .00     .00     .00
  2 DREI              52.00  177.00   20.00   20.00    5    0    0     .00     .00     .00     .00
  3 INNEN            121.00  121.00    7.00    7.00    9    0    0     .00     .00     .00     .00
  4 KARO             110.00  110.00   29.00   30.00   13    0    0     .00     .00     .00     .00
  5 PARA             167.00   50.00   39.00   59.00   22    0    0     .00     .00     .00     .00
  6 KREIS            174.50  174.50   25.00   25.00   25    0    0     .00     .00     .00     .00
STOP    ENDE
```

Figure 8.3: The regions of map MERK.

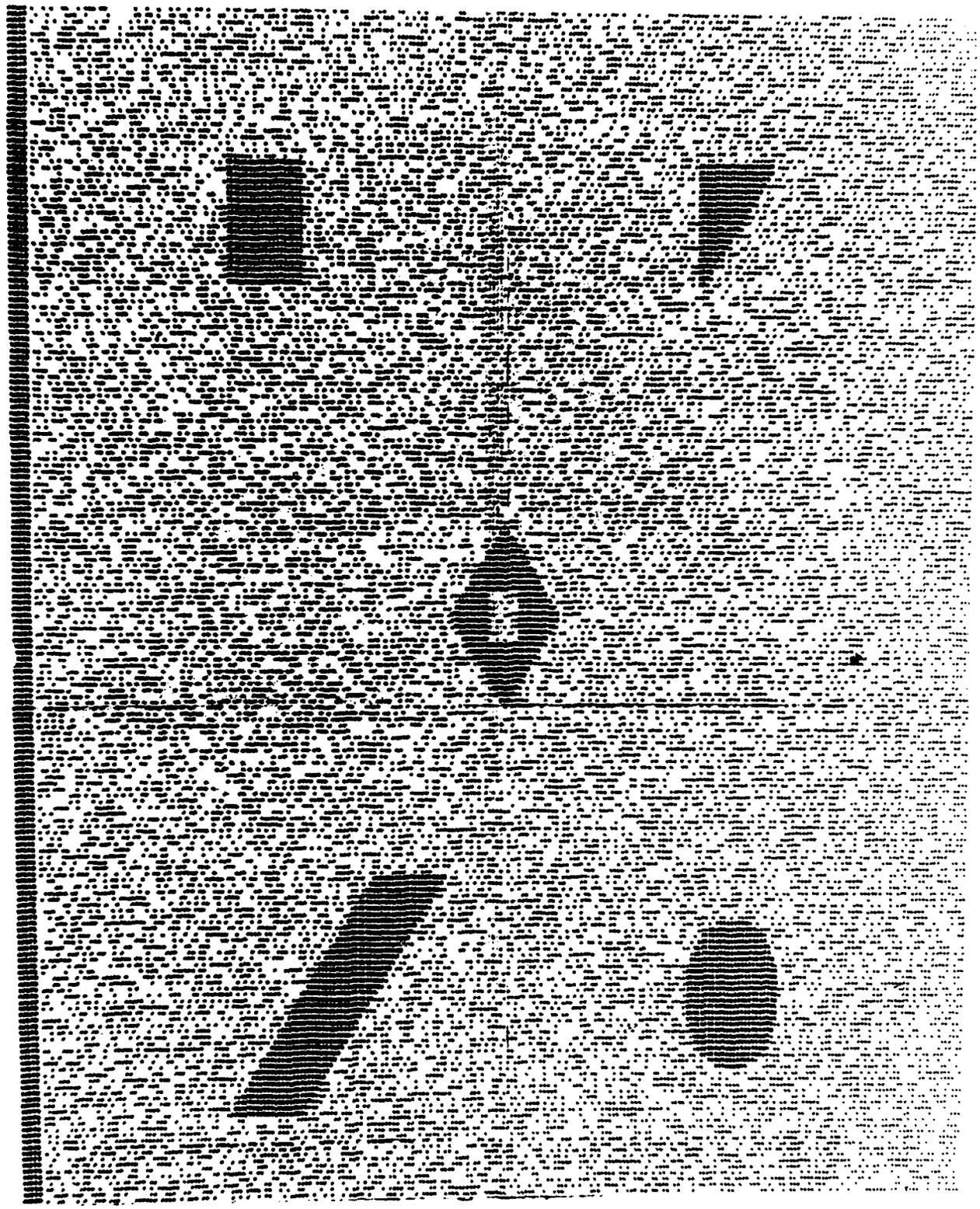


Figure 8.4: Synthetic image MERK with noise.

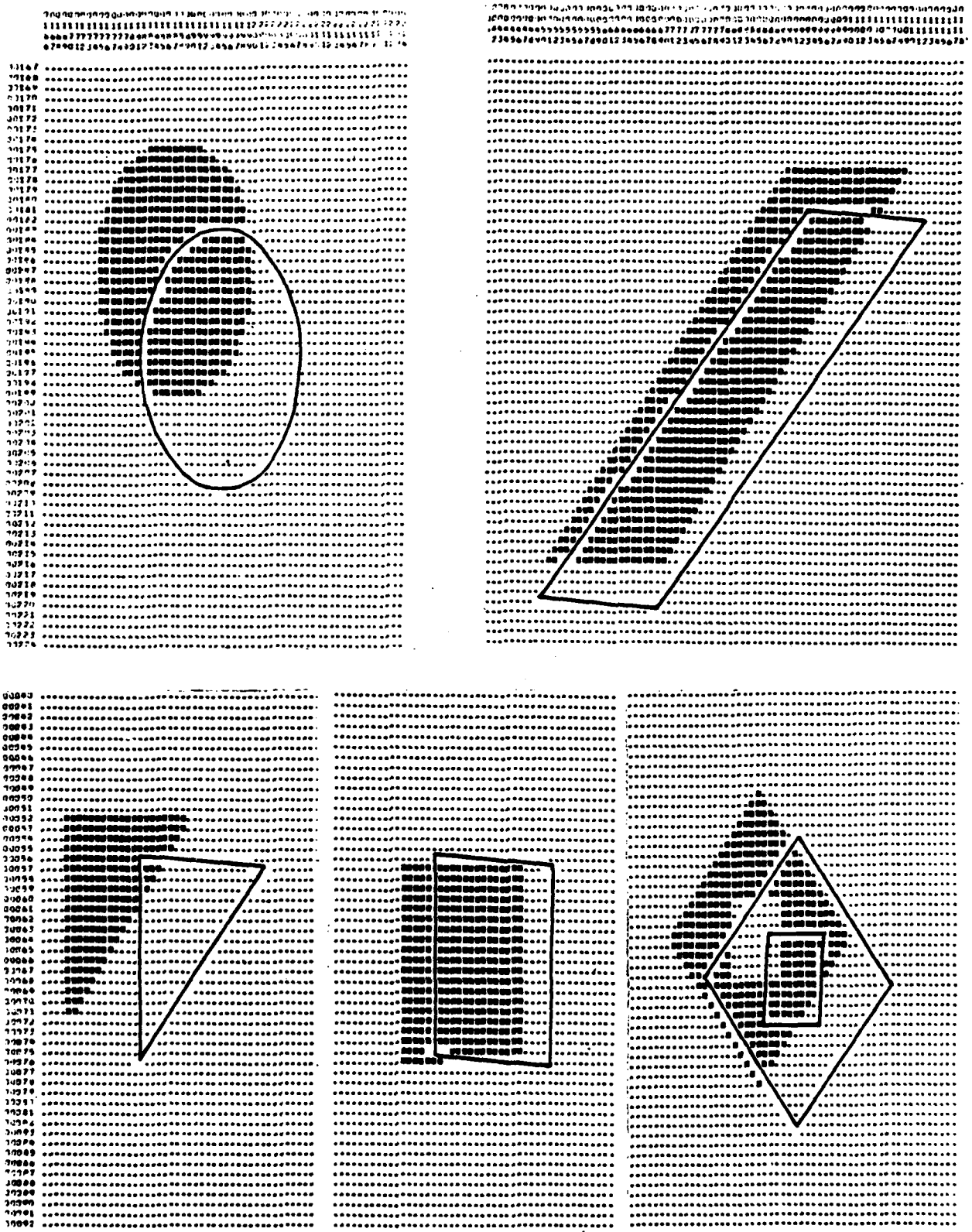


Figure 8.5: Projection of 5 objects from map MERK to image MERK. MERK.

8.2.3 Results with procedure SHIFT

Procedure SHIFT was applied as discribed in chapter 6.2. For all objects a position was found of maximum correlation between the binary map-derived image and the window in MERK.

A section of the matrix containing the positional correlation measures in percent is shown in figure 8.6. The correlation value grows from 14.2 % at the original position until 88.6 % at the maximum (the correlation measure was explained in equ. 6.7). 1ØØ % is not reached because procedure SHIFT takes into consideration only the shift parameters of the projection, however, not rotation and scale factors which cause the deviation of the correlation value from the ideal value of 1ØØ %.

Procedure SHIFT was applied also to the noisy version of image MERK. The maximum correlation was practically the same or even higher in some instances in the noisy image. This can be explained by the smoothing effect of the noise.

79.6	85.5	76.5	65.5	52.2				
85.8	**88.6**	82.3	67.9	54.2	41.2			
77.2	82.3	74.1	65.1	51.9	39.3	27.4		
			57.5	49.3	37.1	25.6	14.8	
				42.4	34.9	23.8	14.1	5.9
					28.7	22.7	**14.2**	7.1
						19.2	15.4	9.5

Figure 8.6: Correlation measures for object KARO in %, and path of increasing correlation from a starting value to the maximum.

This noise can compensate for the effect of the rotation error.

Figure 8.7 compares the results from both images with the aid of error vectors of an overdetermined transformation. The control points used for the transformation are the centers of gravity of the features in the map and the shifted features in the projected version of the map.

To qualify the method a standard deviation for coordinate errors and point errors (vector lengths) are calculated using:

$$\sigma = \sqrt{\sum_{i=1}^{n} d_i/(n-3)}$$

where d_i are the individual residuals after the transformation, n is the number of points; (n-3) is the degree of freedom (overdetermination) of the transformation.

	C l e a n			
Feature	x y	Error vector Δx Δy	Length	Correl. [%]
RECHT	(61.5ØØ , 61.52Ø)	(Ø.692 , -Ø.157)	Ø.7Ø9	87.23Ø
DREI	(58.667 ,183.667)	(-Ø.76Ø , Ø.347)	Ø.835	72.783
KARO	(124.5ØØ ,124.949)	(Ø.178 , -Ø.4ØØ)	Ø.437	88.575
PARA	(186.5ØØ , 79.5ØØ)	(-Ø.887 , Ø.4Ø4)	Ø.974	88.969
KREIS	(187.ØØØ ,187.ØØØ)	(Ø.777 , -Ø.194)	Ø.8ØØ	88.765
	Stand. deviation	(1.113 , Ø.5Ø3)	1.221	85.264
	N o i s e			
RECHT	(61.5ØØ , 61.5ØØ)	(Ø.692 , -Ø.Ø73)	Ø.696	87.16Ø
DREI	(58.667 ,183.667)	(-Ø.76Ø , Ø.22Ø)	Ø.791	72.855
KARO	(124.5ØØ ,124.949)	(Ø.178 , -Ø.3Ø7)	Ø.355	88.592
PARA	(186.5ØØ , 79.5ØØ)	(-Ø.877 , Ø.255)	Ø.923	88.878
KREIS	(187.ØØØ ,187.ØØØ)	(Ø.777 , -Ø.Ø95)	Ø.782	88.777
	Stand. deviation	(1.113 , Ø.333)	1.162	85.252

Figure 8.7: Table of results from procedure SHIFT.

The comparison of the clean and noisy data shows that maximum correlation appears always at the same position except for object DREI. Object DREI finds maximum correlation at the border of the area of interest, consisting of 21 x 21 pixels in a square around the original position. The absolute maximum would lie outside this window. In the clean version two adjacent positions get the same maximum value while in the other version noise influences the correlation. Hence the maximum is found in only one position. This is reflected in the change of one coordinate of the control points of DREI and explains why the error vectors are not identical in the two versions.

8.2.4 Results with procedure THRESH

In the same way as procedure SHIFT, procedure THRESH was applied to the two versions of image MERK with projection (8.1).

The parameters chosen for this example were distance D = 1Ø and factor p = 1.Ø (compare chapter 6.3). Two restrictions to the threshold bounds IPU and IPO were necessary:

(a) The lower bound IPU was fixed to Ø and

(b) "bad" grey values were not allowed by setting IPO < 255.

Tests have shown that no reasonable results can be obtained without these two restrictions since the overlap region between projection and image feature is so small. An alternative has been to choose factor p so small that p*F(M) is less than the surface of the overlap. This works for the clean version but not with noise.

The two conditions above seem to be reasonable because they can be translated into conditions for preprocessing which underlying satellite images should satisfy:

(a) The grey values of the preprocessed image should be low for pixels which belong to the feature and high for the rest.

(b) Pixels with grey value 255 are excluded a priori.

Figure 8.8 summarizes the results of THRESH.

	C l e a n			
Feature	x y	Error vector Δx Δy	Length	Correl. [%]
RECHT	(61.5ØØ , 61.5ØØ)	(Ø.223 , -Ø.224)	Ø.316	98.533
DREI	(58.667 ,183.667)	(-Ø.361 , Ø.248)	Ø.438	85.217
KARO	(124.5ØØ ,124.949)	(Ø.298 , -Ø.Ø62)	Ø.3Ø4	92.823
PARA	(186.5ØØ , 79.5ØØ)	(-Ø.421 , Ø.289)	Ø.51Ø	93.439
KREIS	(187.ØØØ ,187.ØØØ	(Ø.261 , -Ø.251)	Ø.363	94.913
	Stand. deviation	(Ø.5Ø7 , Ø.362)	Ø.623	92.8Ø5
	N o i s e			
RECHT	(61.5ØØ , 61.5ØØ)	(Ø.662 , -Ø.178)	Ø.685	89.472
DREI	(58.667 ,183.667)	(-Ø.721 , Ø.268)	Ø.77Ø	43.128
KARO	(124.5ØØ ,124.949)	(Ø.159 , -Ø.195)	Ø.252	65.279
PARA	(186.5ØØ , 79.5ØØ)	(-Ø.842 , Ø.312)	Ø.898	86.673
KREIS	(187.ØØØ ,187.ØØØ)	(Ø.743 , -Ø.2Ø7)	Ø.771	77.387
	Stand. deviation	(1.Ø59 , Ø.375)	1.124	72.388

Figure 8.8: Table of results from procedure THRESH.

Binary correlation is used here for verification. 1ØØ % is not reached for the same reason as for SHIFT. However, better values for the clean image show that thresholding takes care of feature rotation and scale in the image, but verification considers only the shift of the projection.

Results from the noisy image demonstrate the strong noise dependency of thresholding. On the average correlation is 2Ø % smaller than before. Moreover object DREI could not be accepted as recognized due to a correlation value of 43%. Figure 8.9 explains why DREI has not been recognized automatically. There is a very small overlap between the image feature and the projected map element for the determination of the bounds. Hence very much noise is included. The center of gravity of all threshold pixels is therefore displaced so that verification cannot state a similarity.

8.2.5 Results of procedures LISU and LINDET

The left hand side of the parallelogram PARA was used to test line following. The edge operator of Roberts (chapter 5.2.4) was applied to the image (noisy and clean version: see figure 8.1Ø).

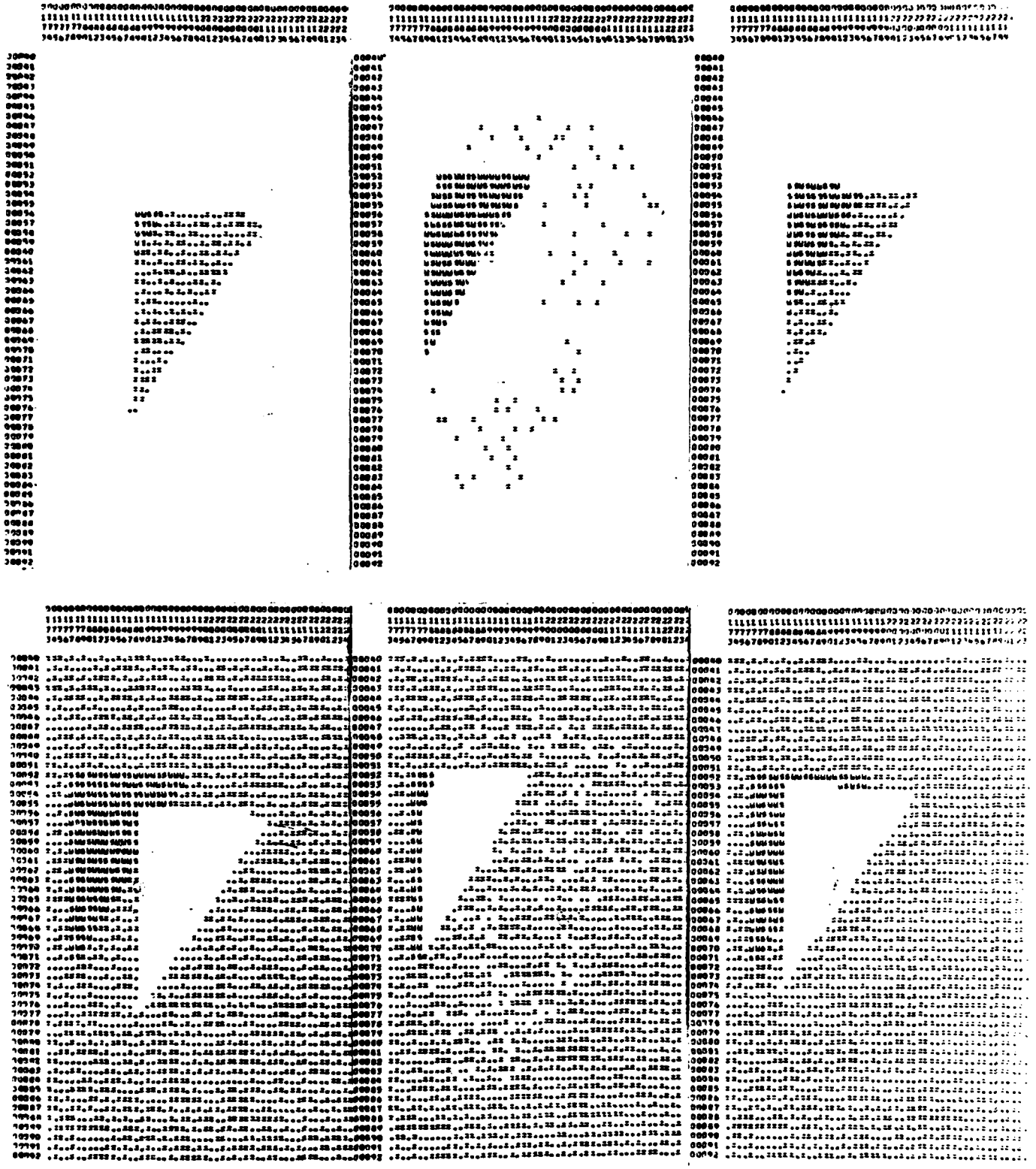


Figure 8.9: Recognition of DREI by thresholding.

(a) original projection (8.1)
(b)
(c) threshold pattern
(d)
(e) verified projection
(f) binary masked printout
complemented binary

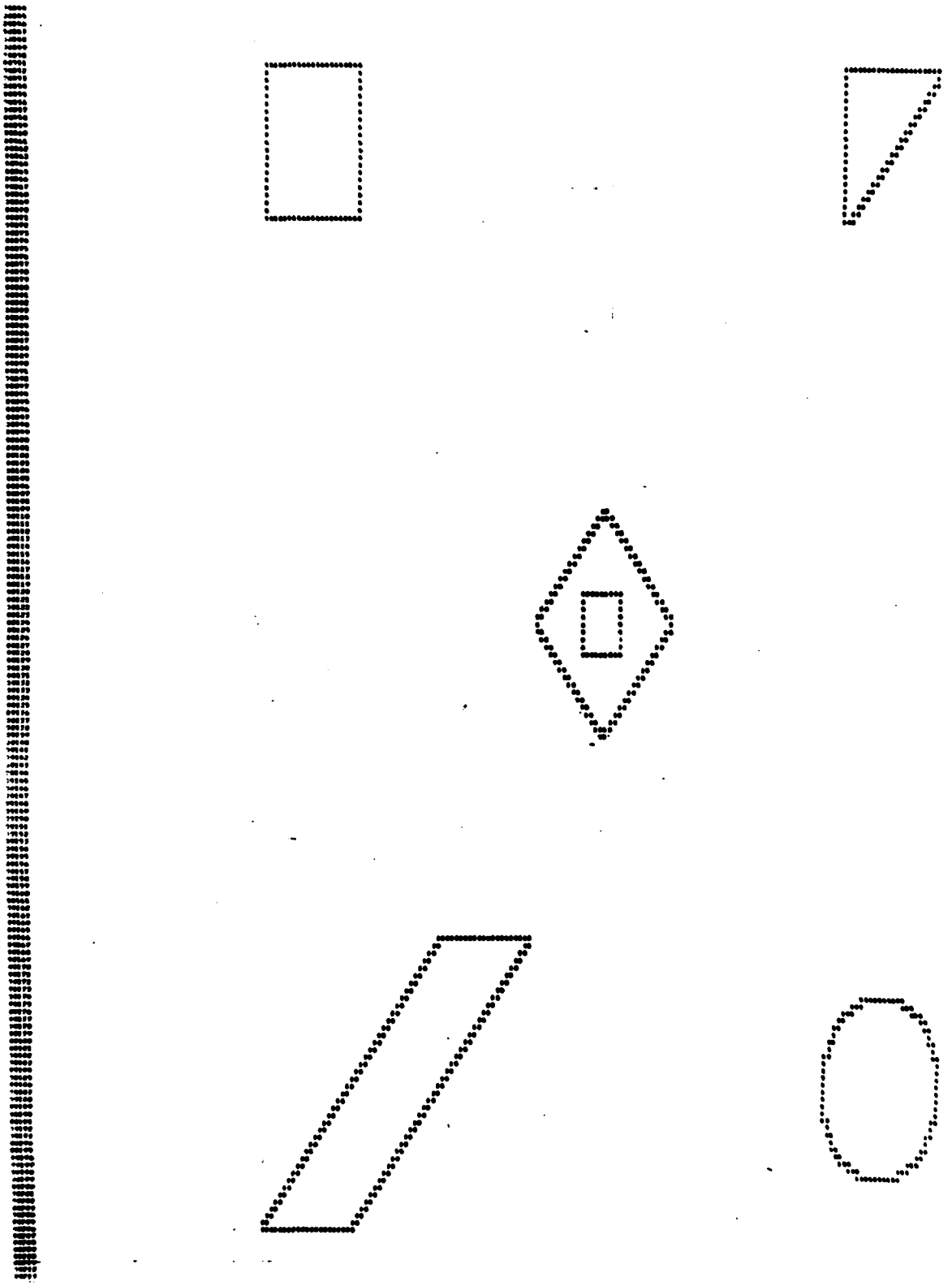


Figure 8.1Ø: Edge image from operator of Roberts.

These edge images were used for line detection. The starting mask was created and the histogram provided the following parameters of the cost-function (see chapter 6.5.3):

Clean version	Noisy version
a = 256.Ø	a = 188.52
b = 1.Ø	b = - Ø.7364
n = Ø.78	n = Ø.85

Table 8.2: Parameters of the costfunction for line following.

Starting and end point S and E are projected into the image (S_p,E_p) and improved (S_I,E_I) by the procedure as illustrated in figure 8.11:

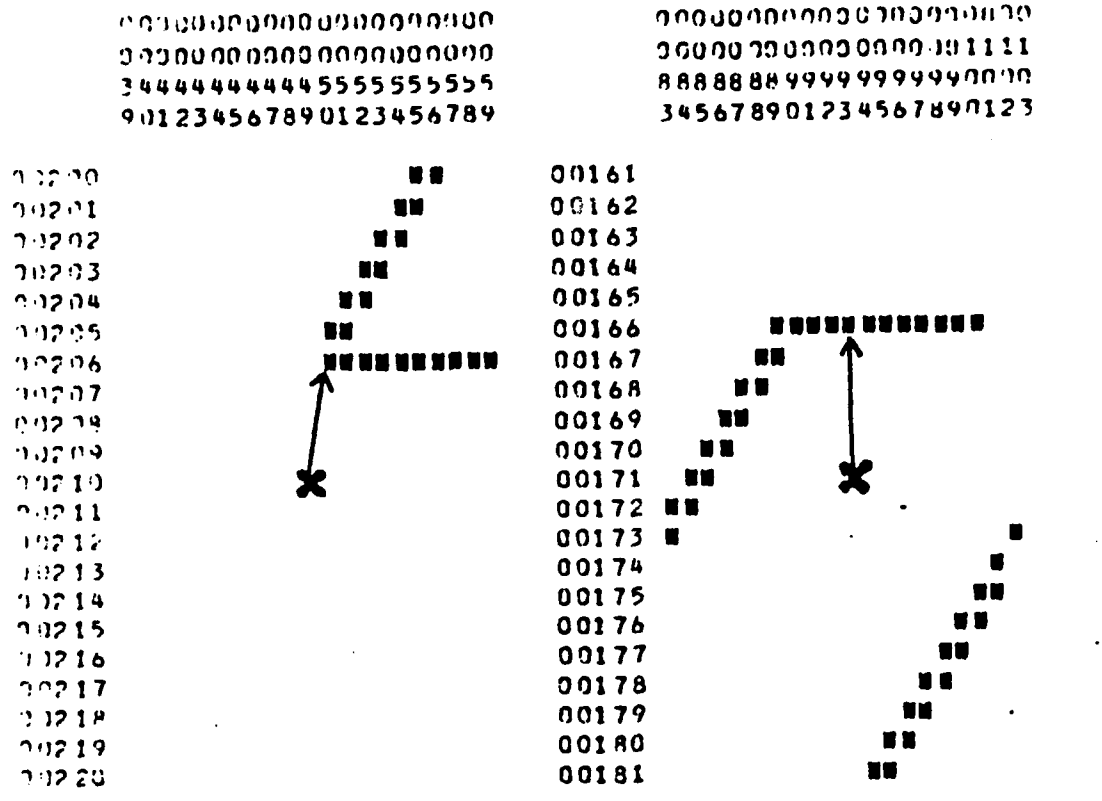


Figure 8.11: Improvement of starting and end point of the line.

(a) (cost S_I) < 1, (cost E_I) < 1

(b) S_I and E_I are contained in the starting mask

(c) Among pixels satisfying conditions (a) and (b) S_I and E_I have the maximum distance to S_p and E_p.

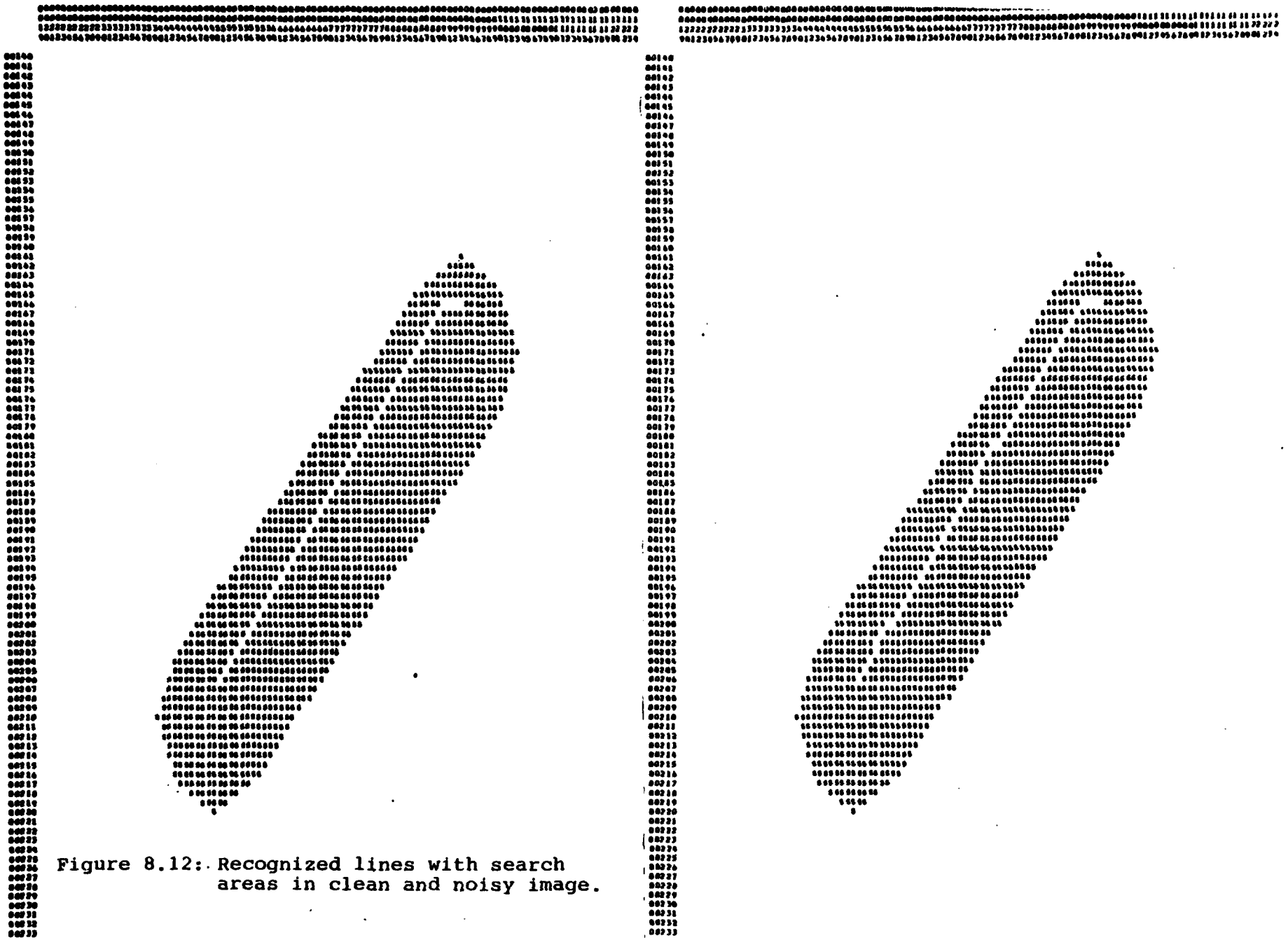


Figure 8.12: Recognized lines with search areas in clean and noisy image.

Black pixels in figure 8.11 are pixels satisfying the first two conditions. The arrow points to the improved extreme points corresponding to condition 3.

Figure 8.12 shows the results of procedures LISU and LINDET within the corresponding search area.

For verification purposes the corresponding lines in the map and image were subdivided into 6 parts of equal length. Subdivision points 2 and 4 are used to calculate a new transformation of best overlay according to chapter 6.6.1(b). All 5 subdivision points of the map are transformed and difference vectors calculated to the 5 corresponding points of the image line. The results are shown in figure 8.13. No errors appear in the clean version because the line is straight. In the noisy picture some pixels do not lie on a straight line. Starting and end points are not included in the verification process because they are not a result of line following.

Point differences of the linear transformation adapted to points 2 and 4					
Subdivision points	Error (clean image) Δx Δy	length	Error (noisy image) Δx	Δy	length
1	(.Ø , .Ø)	.Ø	(.2Ø7 ,	-.2Ø7)	.293
2	(.Ø , .Ø)	.Ø	(.Ø ,	.Ø)	.Ø
3	(.Ø , .Ø)	.Ø	(-.146 ,	.232)	.275
4	(.Ø , .Ø)	.Ø	(.Ø ,	.Ø)	.Ø
5	(.Ø , .Ø)	.Ø	(-.2Ø7 ,	.2Ø7)	.293
Stand.dev.	(.Ø , .Ø)	.Ø	(.231 ,	.264)	.352

Figure 8.13: Verification of the line in MERK.

8.2.6 Results with procedure ADAPT

Considerations of computing time caused us to use a new approximate projection which is improved and brings the objects nearer to the image features. This projection (8.2) has the following parameters

$$T_x(x,y) = -Ø.Ø1 * x + 1.Ø2 * y - Ø.5Ø$$
$$T_y(x,y) = 1.Ø2 * x - 1.ØØ \qquad (8.2)$$

and results in areas for each feature as shown in table 8.3.

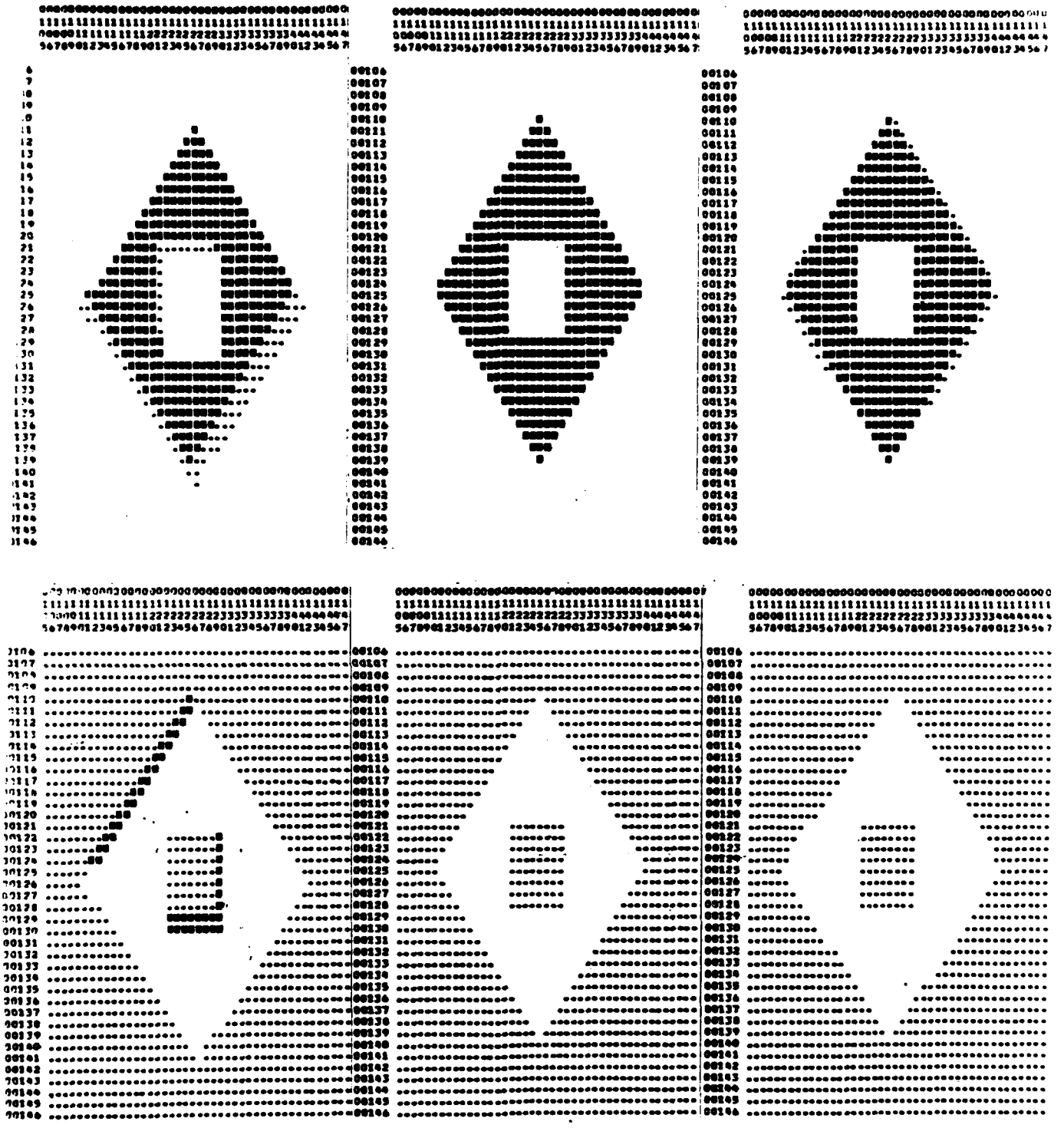


Figure 8.14: Adapting object KARO in the clean image.

(a) original projection (8.2) (b) (c) adapted pattern (d) (e) verified projection (f) binary masked printout complemented binary

Feature	Area orig.	Area proj.	Differences in position(pixels)
RECHT	36.1	40.0	Ø.ØØØ
DREI	2Ø.0	23.2	2.968
KARO	48.4	41.Ø	1.759
PARA	74.1	83.2	2.57Ø
KREIS	48.5	53.9	2.998

Table 8.3: Characteristics of projection 2 in pixel units.

Procedure ADAPT is demonstrated by means of object KARO (figure 8.14). Four iterations have been computed. The changes of the shapes are illustrated in figure 8.15. This figure presents the surface of the feature "binadapt" before each iteration step; the number of pixels that are added (+) and eliminated (-) at each step; and finally the grey value bounds and the number of pixels contained in this interval in binadapt.

Iteration	Starting Area	+	-	Grey value limits: area
1	41Ø	32	48	(Ø,254):334
2	394	2Ø	17	(Ø,254):366
3	397	Ø	11	(Ø,254):386
4	386	Ø	ØØ	(Ø,Ø):386
	386			

Figure 8.15: Shape changes while iterating ADAPT on oject KARO.

During the last iteration step the correct interval (Ø,Ø) is found. Verification, however, is less than 1ØØ % because of orientation and scale differences in projection (8.2) (see figure 8.14e,f).

A summary of results from procedure ADAPT is presented in figure 8.16.

Feature	Position in the image x	y	Error vector Δx	Δy	Length	Correl. [%]
C l e a n						
RECHT	(61.5ØØ ,	61.5ØØ)	(Ø.134 ,	-Ø.Ø1Ø)	Ø.134	1ØØ.ØØØ
DREI	(58.667 ,	183.667)	(-Ø.133 ,	Ø.Ø15)	Ø.134	99.349
KARO	(124.5ØØ ,	124.949)	(Ø.ØØ6 ,	-Ø.Ø11)	Ø.Ø12	95.867
PARA	(186.5ØØ ,	79.5ØØ)	(-Ø.155 ,	Ø.Ø17)	Ø.156	98.533
KREIS	(187.ØØØ ,	187.ØØØ)	(Ø.149 ,	-Ø.Ø11)	Ø.149	98.Ø94
	Stand. deviation		(Ø.2Ø2 ,	Ø.Ø21)	Ø.2Ø3	98.369
N o i s e						
RECHT	(61.5ØØ ,	61.5ØØ)	(Ø.171 ,	Ø.Ø44)	Ø.176	1ØØ.ØØØ
DREI	(58.667 ,	183.667)	(-Ø.127 ,	-Ø.Ø58)	Ø.139	83.6Ø4
KARO	(124.5ØØ ,	124.949)	(-Ø.Ø82 ,	Ø.Ø32)	Ø.Ø88	83.733
PARA	(186.5ØØ ,	79.5ØØ)	(-Ø.148 ,	-Ø.Ø68)	Ø.163	86.561
KREIS	(187.ØØØ ,	187.ØØØ)	(Ø.186 ,	Ø.Ø5Ø)	Ø.193	87.343
	Stand. deviation		(Ø.233 ,	Ø.Ø82)	Ø.247	88.248

Figure 8.16: Results from procedure ADAPT.

For object RECHT projection (8.2) results in an errorfree overlap which is detected by ADAPT in the first iteration. Noise decreases the correlation by about 1Ø %. The effect of noise in the control points is very small.

8.2.7 Comparison of recognition procedures

After the determination of control points for every object with every of the investigated methods an overdetermined 6-parameter linear transformation is computed with residuals to compare the results. Figure 8.17 presents the results from the clean and noisy image.

method	feature	c l e a n error vector	length	n o i s e error vector	length
SHIFT	RECHT	(Ø.45Ø , -Ø.657)	Ø.796	(Ø.9Ø5 , -Ø.567)	1.Ø68
	DREI	(-1.712 , 1.235)	2.111	(Ø.117 , 1.735)	1.739
	KARO	(Ø.Ø83 , -Ø.517)	Ø.524	(Ø.8Ø8 , Ø.175)	Ø.827
	PARA	(-Ø.229 , -Ø.55Ø)	Ø.595	(-Ø.434 , -Ø.Ø74)	Ø.44Ø
	KREIS	(Ø.833 , Ø.Ø58)	Ø.835	(1.816 , 1.34Ø)	2.257
THRESH	RECHT	(Ø.47Ø , -Ø.181)	Ø.5Ø4	(Ø.ØØ9 , -Ø.275)	Ø.289
	DREI	(-Ø.655 , Ø.344)	Ø.74Ø	(-3.396 , -Ø.87Ø)	3.5Ø6
	KARO	(Ø.323 , -Ø.294)	Ø.437	(-Ø.982 , -1.ØØ8)	1.4Ø7
	PARA	(-Ø.139 , -Ø.25Ø)	Ø.286	(-Ø.7Ø6 , -Ø.278)	Ø.759
	KREIS	(Ø.Ø7Ø , -Ø.757)	Ø.76Ø	(-Ø.947 , -1.723)	1.966

Figure 8.17: Overall results with synthetic image MERK (cont'd).

ADAPT	RECHT	(Ø.47Ø , -Ø.181)	Ø.5Ø4	(Ø.925 , -Ø.Ø91)	Ø.93Ø
	DREI	(Ø.665 , -Ø.138)	Ø.679	(1.131 , Ø.535)	1.251
	KARO	(Ø.383 , -Ø.223)	Ø.443	(Ø.78Ø , -Ø.284)	Ø.83Ø
	PARA	(-Ø.139 , -Ø.25Ø)	Ø.286	(Ø.382 , -1.184)	1.244
	KREIS	(Ø.562 , -Ø.265)	Ø.621	(1.152 , -Ø.452)	1.237
LINDET	PUNKT 2	(-Ø.383 , Ø.789)	Ø.877	(-Ø.674 , 1.Ø19)	1.222
	PUNKT 4	(-1.Ø51 , 1.835)	2.115	(-Ø.966 , 2.ØØ1)	2.222
Stand.deviation		(Ø.71Ø , Ø.738)	1.Ø24	(1.322 , 1.1Ø5)	1.723

Figure 8.17: Comparison of recognition methods by error vectors.

For lines, the subdivision points 2 and 4 are used as control points. In the clean version there are two points with error vector lengths greater than 1: DREI with SHIFT and point 4 with LINDET. DREI was the object from which the projection (8.1) was more than 1Ø pixels apart from the image feature. The error of the line subdivision point 4 has its explanation in the fact that the end point E_I in the image lies on the short side of the parallelogram I_{PARA} while the starting point was found correctly. So the line in the image must be longer than the real projection of the map. The effect of that is that the control points are shifted towards the end point.

From the point of view of noise all errors of all procedures become greater but THRESH seems to be the worst. Errors of LINDET are of the same type as in the clean image. Errors of ADAPT seem to be the most homogeneous (Ø.2 to Ø.7 with clean data, Ø.8 to 1.2 with noise) with little variations while the other methods vary more in their performance.

Figure 8.18 summarizes the standard deviation of all methods and the mean of the correlation measures used for verification.

method	noise	standard deviations σ_x	σ_y	σ_p	mean correlation
SHIFT	no	1.113	Ø.5Ø3	1.221	85.264 %
	yes	1.113	Ø.333	1.162	85.252 %
THRESH	no	Ø.5Ø7	Ø.362	Ø.623	92.8Ø5 %
	yes	1.Ø59	Ø.375	1.124	72.388 %
ADAPT	no	Ø.2Ø2	Ø.Ø21	Ø.2Ø3	98.369 %
	yes	Ø.233	Ø.Ø82	Ø.247	88.248 %
LINDET	no	Ø.ØØØ	Ø.ØØØ	Ø.ØØØ	1ØØ.ØØØ %
	yes	Ø.231	Ø.264	Ø.352	98.278 %

Figure 8.18: Comparison of recognition methods by statistical measures.

The correlation measure licorr of the line is calculated from the error vector lengths li with the following formula:

$$licorr = 100 \left(1 - \left(\sum_{i=1}^{n} li\right) \Big/ (n*D)\right)$$

where n is the number of intersection points and D is the maximum shift lengths as assumed.

This table shows again that method THRESH is most sensitive to noise. In general it is better than shift and has its greatest advantage in computation speed. Speed was the handicap of procedure ADAPT in our tests. However, the current version of ADAPT provides plenty of possibilities and parameters to accelerate speed in an improved version.

8.3 EXPERIENCES WITH LANDSAT IMAGE WALCH

Image WALCH was the first satellite image treated. Basic concepts of the recognition methods were implemented and tested and often corrected. The purpose was to obtain an understanding of weaknesses, not, however, to compare the methods.

8.3.1 Image WALCH and map

Figure 8.19 shows the portion of a satellite image from Southern Germany in its 4 spectral bands MSS4, MSS5, MSS6 and MSS7.

This image was available in a NASA-format. This is significant for the present purpose, since NASA performs a grey value normalisation. Grey values in MSS4, MSS5 and MSS6 range from ø to 127, while in MSS7 it is between ø and 63.

The corresponding map data bank covers the area of the image. Figure 8.2ø shows a plot output of the map. Indicated are some features which were used for recognition.

The features are of different size, texture and grey tone. There are villages (WA-SEE-ORT, Schledorf), there is an island, a forest (Karpfau), grassland (Jochberg, Sachenbach, Wasuedufer, Wasuedufer 2, Wasuedufer 3, Kochelnord) and water bodies (Walchensee, Kochelsee, Karpfsee).

channel 4

channel 5

channel 6

channel 7

Figure 8.19: Image WALCH, a section of a satellite image.

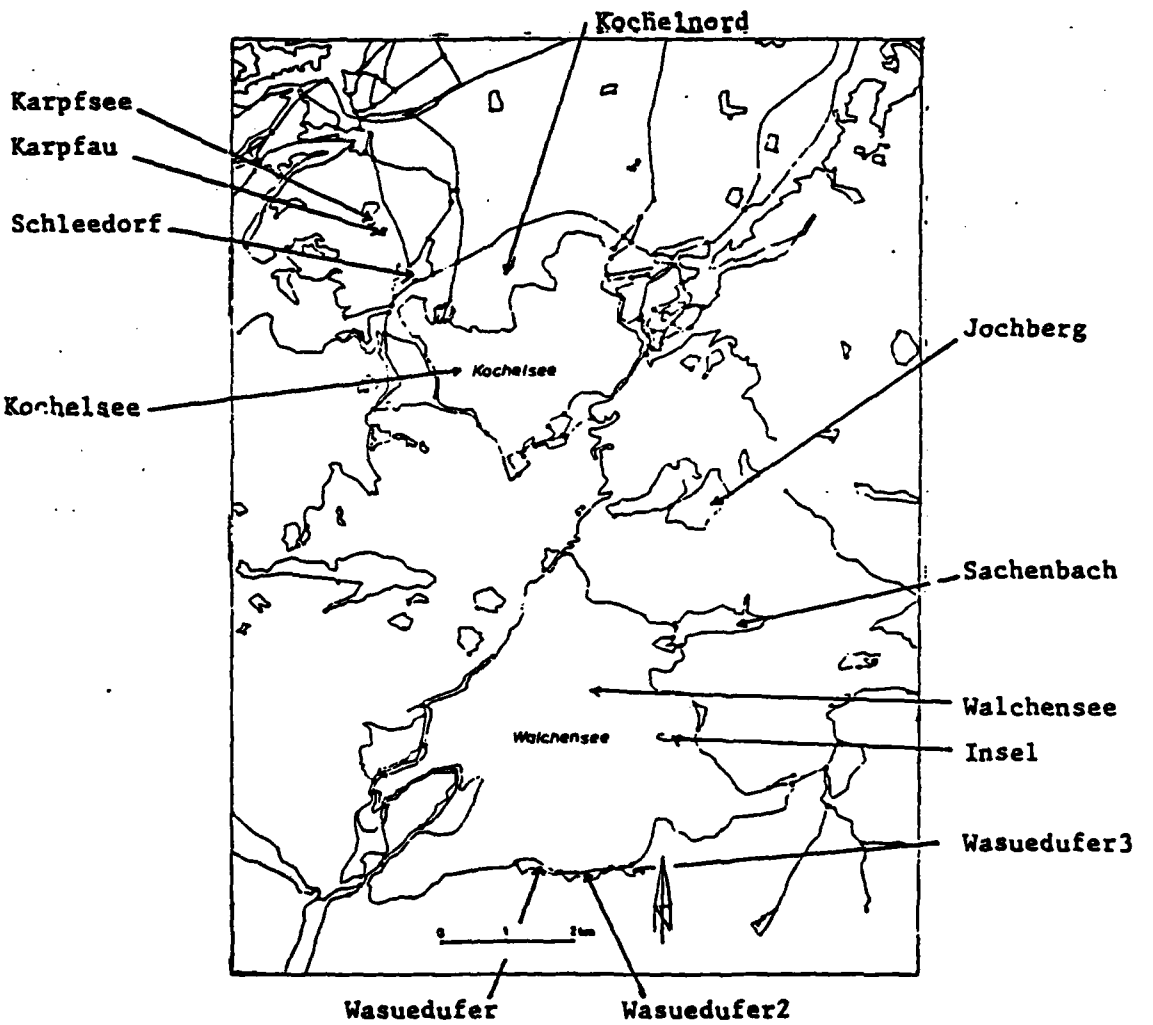


Figure 8.20: Plot of part of the digital map data bank for the test area in Southern Germany. The indicated features were used in the test of the procedure.

Table 8.4 provides a list of objects ordered acc. to size . It could be the result of procedure OBJECTSELECTION (chapter 2.6). The table also lists the centers of gravity of each object which are needed during the overlap process and the type of each region (1 = water body, 3 = villages, 4 = grassland, 5 = forest, 9 = island).

Feature	Area in pixels	Center of Gravity x y	Type
WALCHENSEE	3589	(15Ø2.268,498.Ø28)	1
KOCHELSEE	1374	(15Ø1.242,51Ø.522)	1
KOCHELNORD	5Ø3	(15ØØ.632,512.897)	4
SCHLEDORF-UF	339	(1497.685,51Ø.566)	4
KAKOPFWALD	176	(1497.627,496.215)	5
WA-SEE-ORT	146	(1496.879,498.143)	3
SACHENBACH	126	(15Ø6.761,5Ø1.946)	4
JOCHBERG	113	(15Ø6.3Ø1,5Ø5.659)	4
WALD-1Ø	89	(1496.127,494.969)	5
SCHLEDORF	63	(1497.593,512.858)	3
KAKOPFFREI	32	(1498.525,496.517)	4
WA-SUEDUFER	3Ø	(15Ø1.248,494.392)	4
WA-SUEDUFER 2	28	(15Ø2.8Ø9,494.141)	4
KARPFSEE	21	(1496.2Ø3,514.633)	1
INSEL	14	(15Ø5.347,498.391)	9
WA-SUEDUFER 3	11	(15Ø4.169,494.278)	4
KARPFAU	3	(1496.646,514.131)	5

Table 8.4 : Sorted objectlist for map WALCH.

8.3.2 Projecting the map into the image

An approximate geometric relationship must be established prior to the automated recognition process. The map data are therefore projected into the image with an approximate a-priori 6-parameter transformation. This approximation may, for example, derive from the satellite orbit data.

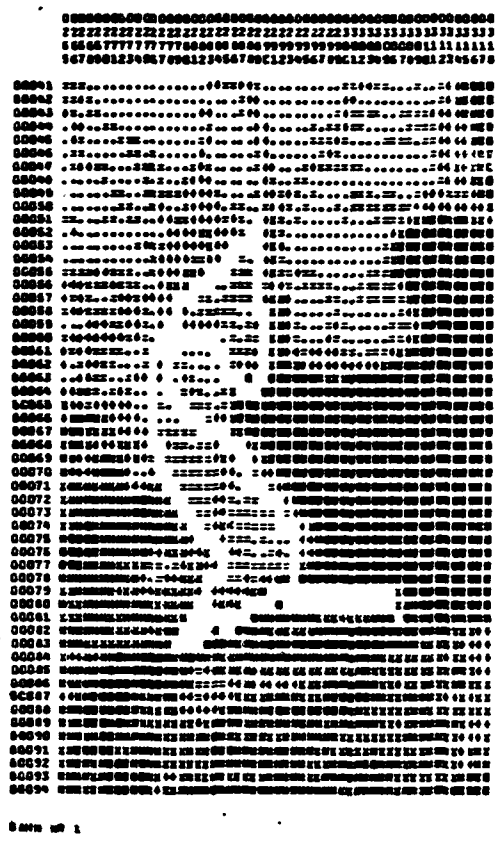
The result of the transformation is shown in figure 8.21. The transformation used the following parameters:

$$T_x(x,y) = 8.213448 * x - 2.6Ø549 * y - 1Ø685.4$$
$$T_y(x,y) = -1.489149 * x - 6.12Ø147 * y + 5423.8 \qquad (8.3)$$

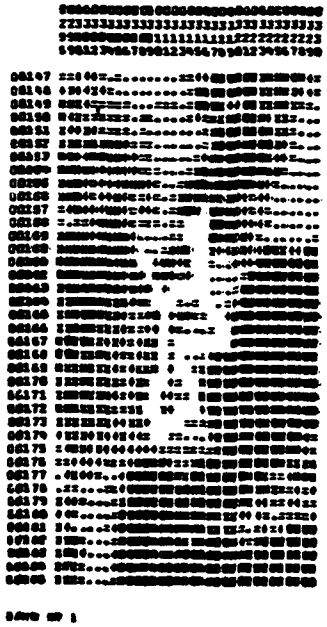
(a) Walchensee

Figure 8.21: Projection of objects WALCHENSEE, KOCHELSEE, KAKOPFWALD, SCHLEDORF-UF and QALD-1Ø into image WALCH (cont'd).

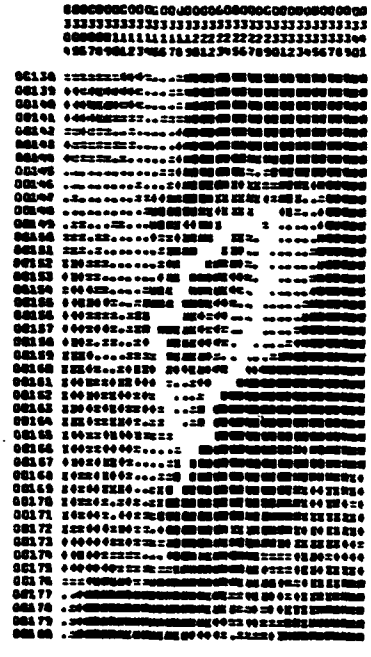
(b) Kochelsee



(c) Kakopfwald



(d) Schledorf-UF



(e) Wald-1Ø

Figure 8.21: cont'd.

8.3.3 Results from procedure SHIFT

Procedure SHIFT was applied to 24 objects of the map. 13 could be recognized with a correlation greater than 5Ø %. Figure 8.22 lists the names of objects, the channels for which maximum correlation was reached, the correlation measures in percent and error vectors with their lengths.

Feature	Error vector x	y	Length of vector	Channel	Correlation
WALCHENSEE	(-Ø.69Ø ,	Ø.161)	Ø.7Ø8	7	6Ø.658
KOCHELSEE	(-1.1Ø7 ,	-Ø.398)	1.177	7	73.1Ø2
KARPFSEE	(1.682 ,	-Ø.162)	1.689	6	55.519
WA-SEE-ORT	(-1.317 ,	Ø.63Ø)	1.46Ø	4	55.Ø57
SCHLEDORF	(-1.328 ,	-Ø.285)	1.358	4	67.277
WA-SUEDUFER	(Ø.714 ,	Ø.425)	Ø.831	7	61.29Ø
WA-SUEDUFER 2	(2.Ø54 ,	Ø.319)	2.Ø79	5	72.195
WA-SUEDUFER 3	(-Ø.943 ,	-1.767)	2.ØØ3	5	58.777
JOCHBERG	(-Ø.166 ,	Ø.379)	Ø.414	4	79.698
KOCHELNORD	(Ø.Ø2Ø ,	Ø.492)	Ø.492	5	6Ø.Ø84
SACHENBACH	(Ø.572 ,	Ø.49Ø)	Ø.753	4	69.8ØØ
KARPFAU	(Ø.72Ø ,	-Ø.258)	Ø.765	5	69.479
INSEL	(-Ø.211 ,	-Ø.Ø26)	Ø.213	7	6Ø.999
Stand.deviation	(1.2Ø7 ,	Ø.692)	1.391		

Figure 8.22: Summary of results from procedure SHIFT.

The distribution of the projected objects and the direction in which they are shifted are shown in figure 8.23. The control points are used for a subsequent image rectification which is described in chapter 8.3.7.

8.3.4 Results from procedure THRESH

Thresholding itself is a very simple algorithm. Difficulties arise for the choice of threshold bounds. How could they be determined automatically? The only aid is the histogram if no external information is available about grey values. Even some rough estimations must be based on a histogram of grey values in the area of interest.

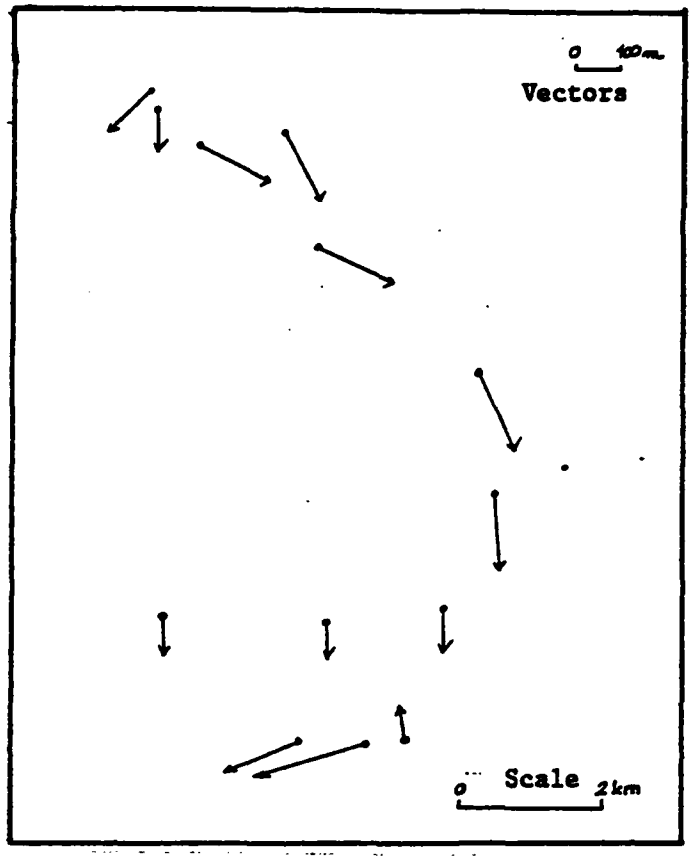


Figure 8.23: Deformation vectors as obtained from the recognized 13 features of emthod SHIFT.

There are two possibilities to vary parameters:

(a) The selection of a region can be varied, thus the set of pixels which is considered for the histogram (histogram mask).

(b) The algorithm can be varied to determine from a given histogram the threshold bounds.

The following histogram masks were used:

H1(D): The rectangular window enclosing the map object as projected into the image, plus a border of D pixels width in each coordinate direction.

H2(D): The projection of the object minus a band of D pixels width from the border.

H3(D): The projection of the object plus a band of D pixels width from the border.

The following grey value boundary determination algorithms were used (IPU = lower bound, IPO = upper bound):

G1 uses the mean M and standard deviation S:

IPU = M - S , IPO = M + S,

G2 determines IPU and IPO by hand,

G3(p) equals procedure ADAHIS with parameter p (see chapter 6.3).

The final procedure is a product of tests with image WALCH, where these alternatives are weighted against one another.

In the example of lake WALCHEN, we know from the map that the water should be dark on channel 7 of the satellite image. Using the histogram over H1(9), and the known surface of the feature with G3(1.3), a thresholding operation produces a new image (TBI) with pixels that are predicted to include the feature. Many other pixels are, however, also included (figure 8.24a). In using map-guidance the binary image OBI (figure 8.21a) serves as a template to eliminate the noisy pixels and to obtain a result, ABI, as shown in figure 8.24b.

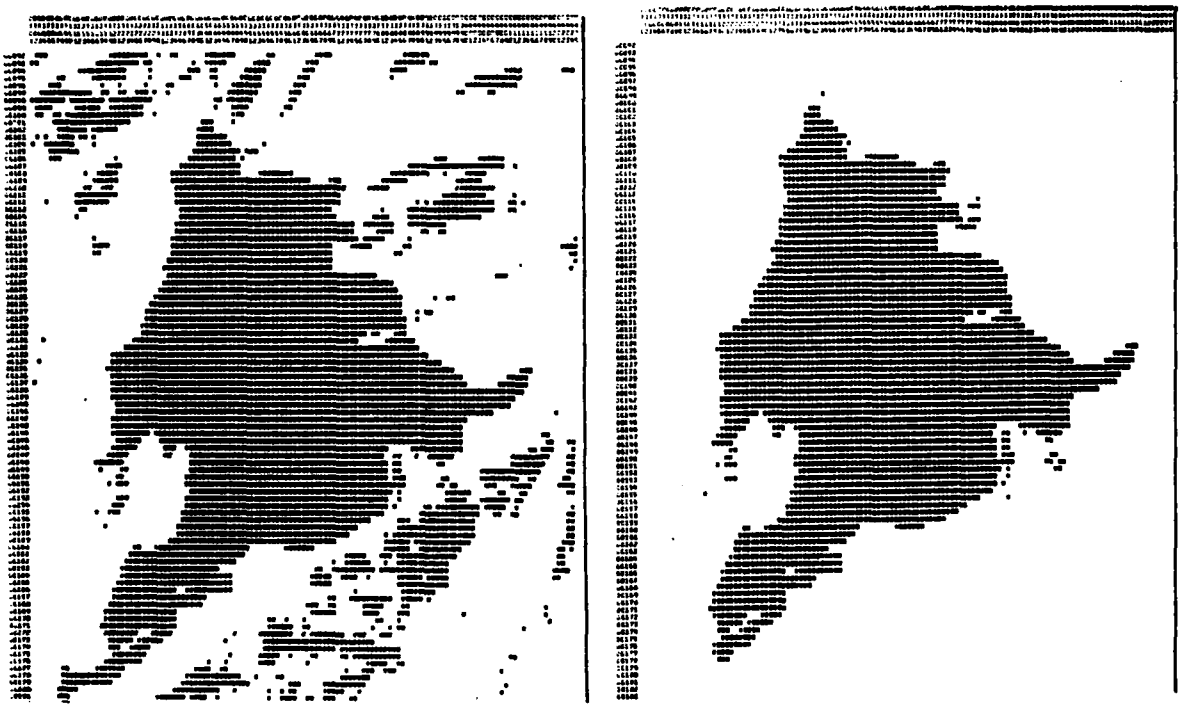


Figure 8.24: Use of thresholding to collect pixels of lake WALCHEN: (a) raw thresholding of a slice of pixels from the histogram, (b) noisy pixels are eliminated using the map feature as a template.

The sequence thresholding-elimination of noisy pixels via map-guidance was inverted in the proposed procedure according to chapter 6.3. The noise elimination can be performed using a distance transform or can be substituted by the preselection of the starting mask.

Further results of different constellations for threshold band determination are presented in figure 8.25 using 4 objects.

Object	Thresh. bounds	Correl.	Diff. vector to shift res.
WALCHENSEE			
1: H1(1Ø),G1	(1,26)	54.735 %	(-8.6 , -1.8)
2: H1(1Ø),G3(1.Ø)	(Ø, 5)	88.282 %	(Ø.5 , Ø.3)
3: H3(5)*,G3(Ø.86)	(Ø, 5)	86.882 %	(3.6 , Ø.2)
4: H1(9),G3(1.)	(Ø, 4)	91.Ø6Ø %	(Ø.7 , Ø.9)
5: H1(9),G2	(Ø, 8)	74.Ø96 %	(-1.9 , -1.3)
+NE**		89.385 %	(-Ø.1 , -Ø.9)
+NE**		93.752 %	(1.Ø , Ø.1)
KOCHELSEE			
1: H1(1Ø),G1	(5,37)	21.511 %	(2.9 , -5.5)
2: H1(1Ø),G3(1.)	(1, 5)	7Ø.5Ø7 %	(-3.1 , -4.3)
3: H2(5)*,G2	(1, 5)	67.661 %	(Ø.Ø , -Ø.9)
4: H3(5)*,G3(Ø.77)	(1, 5)	8Ø.866 %	(-1.Ø , -2.4)
SCHLEDORF-UF***			
1: H3(5)*,G1	(8,38)	37.965 %	
2: H3(5)*,G3(Ø.81)	(33,45)	51.773 %	
KAKOPFWALD***			
1: H2(5)*,G1	(4,26)	72.ØØ4 %	
2: H3(5)*,G1	(1,34)	44.982 %	
3: H3(5)*,G3(Ø.9Ø)	(1, 4)	41.247 %	

Figure 8.25: Results from procedure THRESH for 4 selected objects.

* Results impaired by exceptional effects in determining the histogram mask. These effects, if existing, could be eliminated by refined methods.

** Elimination of noise after thresholding (like fig. 8.24).

*** Objects not recognized with SHIFT, therefore no difference vector.

The grey tone boundary determination algorithm depends on the histogram masks. G1 could be combined with H2 (see feature KAKOPFWALD,1) but did not work well with H1 (WALCHEN-SEE,1; KOCHELSEE,1) nor with H3 (KAKOPFWALD,2) where the resulting threshold interval is too large. The parameter p of G3(p) depends also on the histogram mask used. With H1, p can be greater or equal to 1 because the mask's surface is much greater than the surface of the projection; with H2 a combination with G3(p) is not meaningful because the surface reduction by H2 eliminates already most of the noise and in this case p depends also on the shape of the object, so that no proportional relationship can be established. Good results were obtained from G3(p) with H3. Parameter p was chosen to be less than 1.Ø to compensate discretisation errors of the object projection.

From all these experiences follows that the version of THRESH in chapter 6.3 could be adequate in most cases.

8.3.5 Test and results with procedure LINDET

The objective of tests was to identify the border line of lake WALCHEN. Border lines can be searched either on original images or on preprocessed images. The two search problems differ only in the choice of a different form of the cost function. The simultaneous search on all 4 channels of the image was found to be impracticable because of currently very long computation times. Hence the 4 channels have been compressed according to equation 5.25 in chapter 5.5. Parameters were calculated with the aid of feature type lake (WALCHENSEE, KOCHELSEE). An image section with lake WALCHEN is shown in figure 8.26.

Figure 8.26: Portion of compressed image (Equ. 5.25) with lake WALCHEN.

Line following was divided into 3 segments, which are searched independently. The following cost function was used.

$$c(x,y) = 255 - |g(y) - g(\text{left}(y))| \qquad (8.4)$$

where y is the pixel on the right hand of the edge and left (y) defines its left hand counterpart. Starting and end points of the three segments are extremal points of the contour WALCHENSEE as designated in figure 8.27. This segmentation is necessary for the Euclidean distance used as heuristic function. The union of the three segments and the corresponding development areas are shown in figure 8.29a.

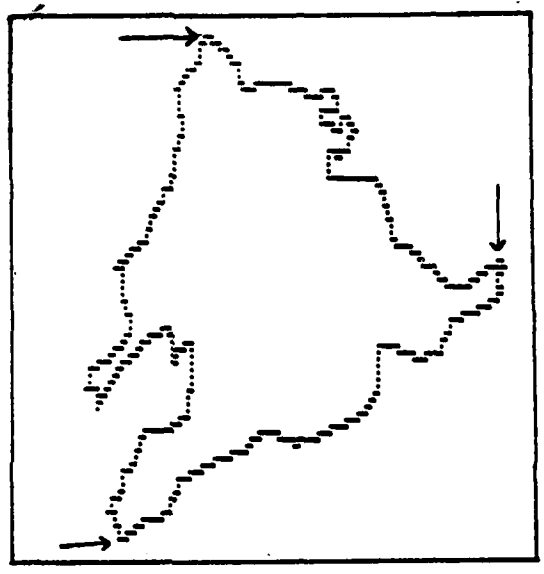


Figure 8.2): Segments of border line of WALCHENSEE, projection from map WALCH.

The second method works on an edge image. The operator of Roberts was applied to the compressed image as shown in figure 8.28.

Figure 8.28: (a) Edge image* of Roberts and (b) binary edge image after thresholding.

Besides the use of an Euclidean distance as a heuristic also the following cost function satisfies the conditions 1,2 and 3 of chapter 6.5.2:

$$cg(x) = (9.6859 - \emptyset.\emptyset37849 * x)^{2.78} \qquad (8.5)$$

The union of the three segments with corresponding development areas is shown in figure 8.29b.

---

* The edge representation uses dark pixels for small edges and white pixels for high edges except Ø edges which are transformed to blanks.

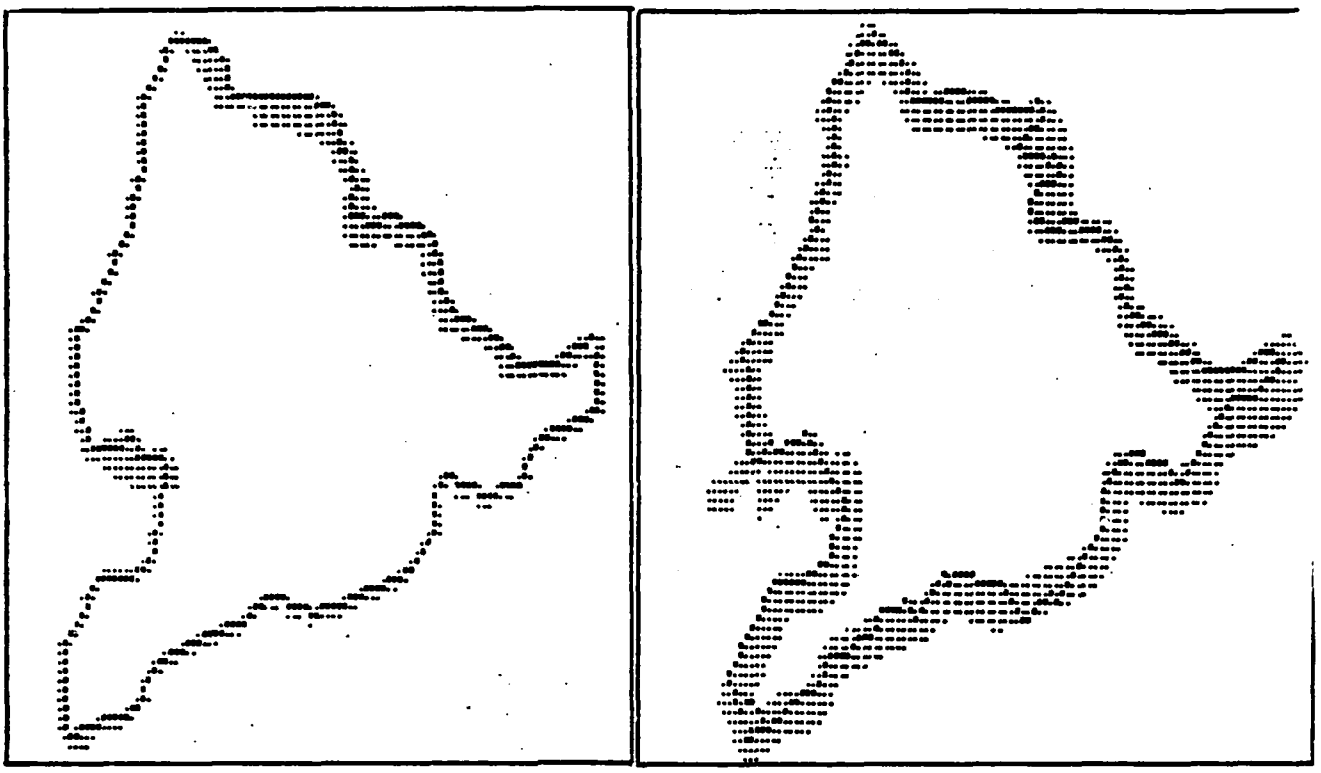


Figure 8.29: Border line of lake WALCHEN with development area:
(a) found in the compressed original image and
(b) found in the edge image.

Although the so-called development area (compare chapter 6.5) in the second case (b) is greater, its completion time was less than in the first case (a). The reason can be the expensive and repetitive computation of edge values for the compressed image, while for the edge image the values are already precalculated. This allows the use of more sophisticated edge operators than in the first case.

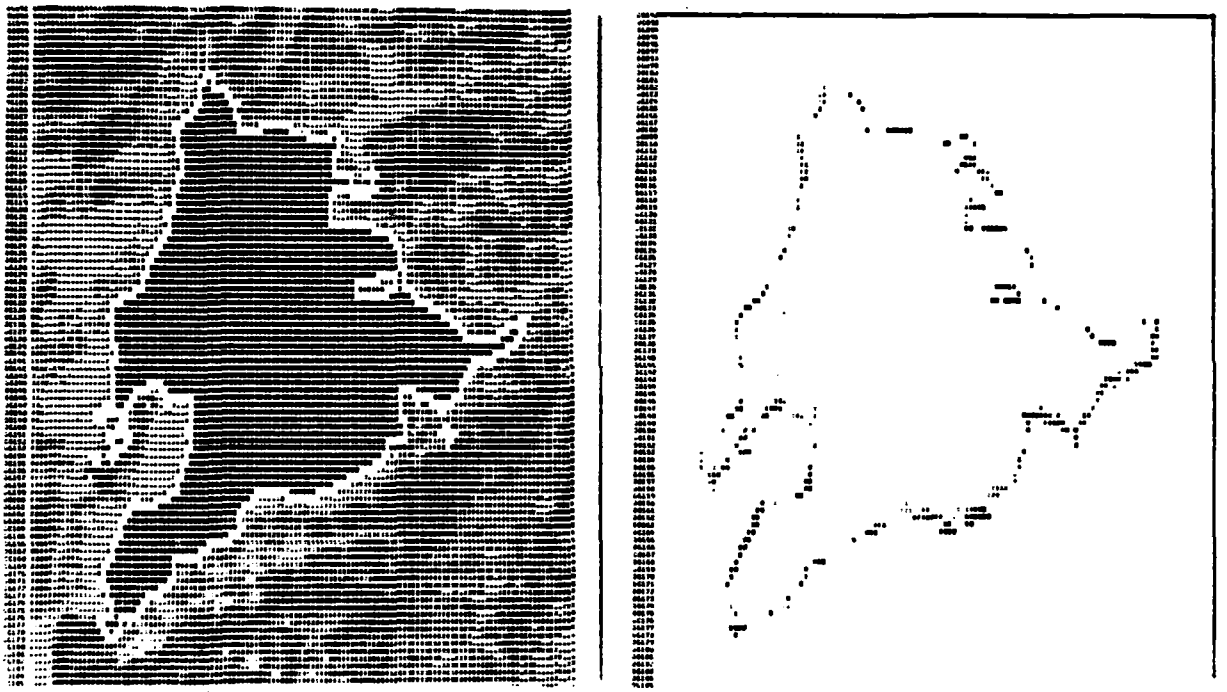
One part of the line has been modified at the left side (compare projected object in figure 8.27). The line follower skipped the small branch of the lake because it is not evident in the image. In general the line from the compressed image is smoother than the line from the edge image which seems to be more sensitive to little changes in direction.

8.3.6 Experiences with procedure ADAPT

As with procedure THRESH, also ADAPT was applied first to object WALCHENSEE.

The determination of the grey level boundary was done in this example by method G1 (see chapter 8.3.4).

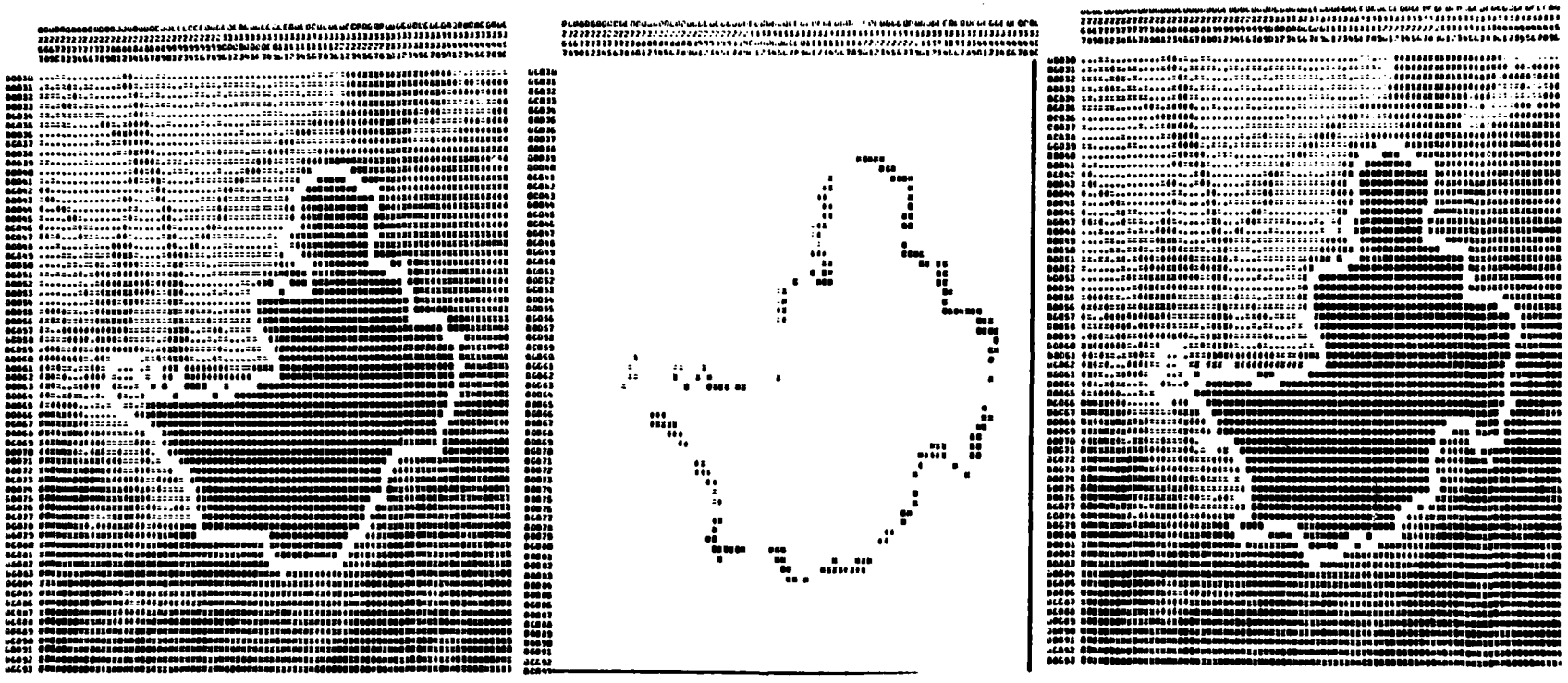
A histogram of the image in figure 8.21 (OBI) will reveal that pixels are included that should be eliminated. On the other hand, pixels in the remainder of the image (in the negative of OBI), are not part of the approximate map-defined feature, but are adjacent to it and should be added. An iterative search along the boundary of the feature will thus permit to correct the image (OBI), using its histogram to decide which pixels to reject and which ones to accept. This "Cinderella-algorithm" produces the result shown in figure 8.3Øa, denoted as image ABI (adapted binary image).



(a) (b)

Figure 8.3Ø: Final image feature after histogram adaption is shown in (a), whereby the boundary is marked with blanks. Figure (b) showes the differences that remain between the corrected image feature and the now correctly transformed map feature.

Further results are described in figure 8.31 regarding the second largest object KOCHELSEE.



(a) (b) (c)

Figure 8.31: Results for the second most probable target for automatic recognition in the test image, lake KOCHEL. The preliminary transformation, using the output from lake WALCHEN, is shown in (a), where the map feature is marked by blanks. Histrogram adaptation changes the pixels inside the map feature as shown in (b), where both added and deleted pixels are printed. Figure (c) shows the final shape of the image feature. Note the differences with respect to the shape of the map feature in (a).

After these early experiences from lakes WALCHEN and KOCHEL we applied ADAPT to seven objects of the map. These objects were searched on channel 7 of the satellite image in two and in three iteration steps. Results are listed in figure 8.32.

Feature	Differences		Threshold boundary		Correlation	
	2. Iteration	3. Iteration	+	-	2. Iteration	3. Iteration
WALCHENSEE	339	4Ø9	(-3,1Ø)	(7,3Ø)	95.239	94.212
KOCHELSEE	181	215	(-3,18)	(12,41)	93.44Ø	92.176
WA-SEE-ORT	52	71	(25,46)	(6,31)	82.Ø11	75.6Ø2
SCHLEDORF-UF	18Ø	23Ø	(2Ø,44)	(4,37)	71.565	63.777
WALD-1Ø	65	65	(15,38)	(3,31)	59.963	57.Ø66
KAKOPFWALD	35	51	(6,36)	(Ø,31)	89.668	84.651
KAKOPFFREI	46	56	(22,48)	(-3,23)	56.575	52.Ø52

Figure 8.32: Results of method ADAPT on MSS7 of image WALCH.

The threshold boundary was determined by method G1 (see chapter 8.3.4) for adding pixels (+) working on a histogram mask H3(Ø) and for determination of pixels (-) working on the complement of H3(Ø). The resulting grey level intervals overlap in some cases very much, so that the same pixel, which has been added in the first step, is eliminated in the second step. Furthermore no grey level adaptation was performed from iteration to iteration. These disadvantages led us to the procedure formulated in chapter 6.4.

Difficulties arose for verification because with increasing iterations the correlation of the adapted feature with the projected decreased in all cases, although the centers of gravity of the projection moves towards the visually perceived collection of feature pixels.With each iteration ADAPT changes slightly the shape of the feature so that it can become less similar to the shape from the map. This can as a rule be the case at the beginning of ADAPT. It can be compensated at higher iterations if in essence the image features is identical in shape to the map feature. A prerequisite for reaching an improved similarity is the sufficiently high number of ADAPT-iterations.

Figure 8.33 has been prepared to show that the deterioration of the correlation measure does not mean a concurrent motion of the image feature into an erroneous direction. It compares ADAPT to SHIFT results, listing the difference vectors between corresponding centers of gravity. While correlation deteriorates, difference vectors do not change significantly.

Feature	After 2 iterations		After 3 iterations	
	Difference vector	Length	Difference vector	Length
WALCHENSEE	(Ø.151 , -Ø.849)	Ø.863	(Ø.391 , -Ø.713)	Ø.813
KOCHELSEE	(-Ø.278 , -1.ØØ4)	1.Ø42	(-Ø.585 , -1.1Ø8)	1.253
WA-SEE-ORT	(-Ø.844 , -Ø.699)	1.Ø96	(-Ø.883 , -Ø.88Ø)	1.247
SCHLEDORF-UF	(1.518 , 1.824)	2.374	(1.7Ø6 , 3.149)	3.582
WALD-1Ø	(-1.6Ø6 , -Ø.9ØØ)	1.841	(-1.695 , -Ø.184)	1.7Ø5
KAKOPFWALD	(-1.4Ø4 , Ø.425)	1.467	(-1.327 , Ø.461)	1.4Ø4
KAKOPFFREI	(-1.725 , -Ø.355)	1.761	(-1.841 , -Ø.423)	1.889

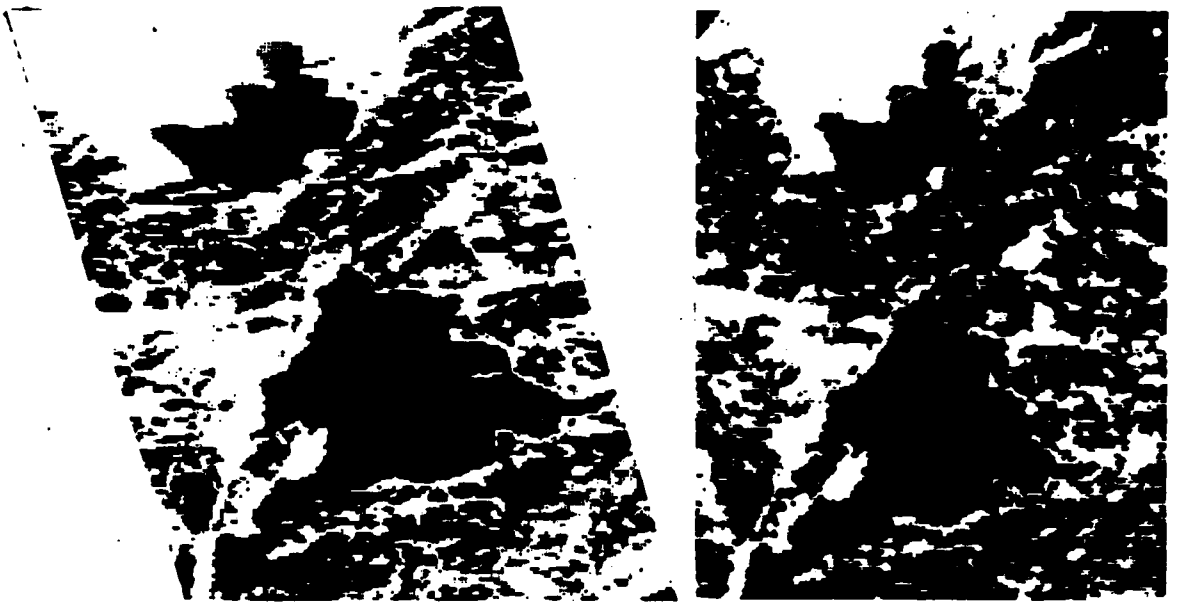
Figure 8.33: Differences between results of SHIFT and ADAPT with 2 and 3 iterations.

For objects SCHLEDORF-UF and WALD-1Ø of type grassland the automatic grey level determination found the wrong intervals. Objects KOCHELSEE and WA-SEE-ORT deteriorate little while objects WALCHENSEE, WALD-1Ø and KAKOPFWALD show decreasing distances to the estimated centers of gravity from procedure SHIFT. The problem of an effective verification could only be resolved by allowing a maximum iteration count much higher than in the present case. Computing time considerations prevented us from more extensive tests.

8.3.7 Rectification of image WALCH

Image WALCH was rectified using 13 control points that were defined automatically with method SHIFT.

The 13 features served to compute centers of gravity. This led then to a set of control points for a rectification. Figures 8.34a,b present the result of the rectification.



(a) (b)

Figure 8.34: LANDSAT-scene from Southern Germany, with raw version in (a) and rectified image in (b), using 13 features.

8.4 RESULTS FROM LANDSAT IMAGE GRAZ

Whereas image WALCH was used to study feature recognition methods and their suitability in connection with automatic map-image registration, image GRAZ had the purpose to test automation and performance of the overall concept.

8.4.1 Image GRAZ and map

The original satellite image channels 4, 5, 6 and 7 of the area of interest chosen for this test were shown in figure 5.15(a)-(d). It is an industrial region with the city of GRAZ, and with river MUR from left top to right bottom. Two highways are marked linear features. A compressed image especially preprocessed for feature type water (river MUR, see figure 5.18) was used for line following. In the same way the four original image channels are compressed for feature types buildings (3), grassland (4) and forests (5). For the highways a ratio image of channels 7 and 4 was used. Figure 8.35 shows a printout of the area of interst.

Figure 8.35: Ratio image (MSS7/ MSS4) of GRAZ.

A plot of the corresponding map was shown in figure 3.2 (chapter 3.2.3). A grid overlay helps to locate map coordinates in that plot. It should be stressed that the digital map data bank was created by an operator who did not work with the purpose of satellite feature recognition. Therefore the map contains political boundaries and other features that can be predicted not to be identifiable from a LANDSAT image .

A list of areal features which have been used for recognition is given in fig. 8.36. This list is sorted automatically according to the object size of the projection in the image and containes further a point of the feature (center of gravity) in map coordinates and following types: 3 for buildings, 4 for grassland and 5 for forests.

Feature	Number of pixel	Type	Center of gravity in map coordinates
WALTENDORF	3118	3	(225.511 , 2Ø46.383)
KAISERWALD-S	3114	5	(218.68Ø , 2Ø16.149)
SCHOTTERGRUB	1712	4	(219.796 , 2Ø31.1Ø3)
EGGENBERG	1429	3	(214.445 , 2Ø49.69Ø)
KAISERWALD-N	1143	5	(213.668 , 2Ø21.387)
ROSENBERG	825	5	(22Ø.635 , 2Ø53.956)
KOLLISCH	8ØØ	5	(231.189 , 2Ø12.443)
MURFELD1	564	4	(223.3Ø6 , 2Ø39.623)
MESSENDORF	538	4	(226.511 , 2Ø41.478)
GEIDORF	5Ø6	3	(221.888 , 2Ø51.61Ø)
WUNDSCHUH	337	5	(224.7Ø8 , 2Ø19.183)
MURFELD2	29Ø	4	(223.465 , 2Ø37.649)
FORST	258	5	(219.416 , 2Ø29.Ø27)
GREITH	2Ø4	4	(23Ø.174 , 2Ø11.791)
PUCHWERK	89	3	(226.224 , 2Ø36.672)
SCHLOSSBERG	48	5	(22Ø.12Ø , 2Ø49.48Ø)
FLIEGERHORST	34	3	(219.9Ø6 , 2Ø29.259)
THALERHOF	19	3	(221.295 , 2Ø31.384)

Figure 8.36: Sorted object list for image and map GRAZ.

8.4.2 Approximate projection from the map to the image.

The projection for image WALCH was approximated manually rather than with satellite orbit data; a set of control points was defined by hand and a transformation was calculated. For image GRAZ the approximate transformation function was calculated totally from informations delivered on the CCT magnetic tape header of the satellite image. The calculation was based on the satellite's position (47°17') and attitude and some specific properties of the LANDSAT-sensor according to descriptions of ANUTA (1973) and MALILA et al. (1973). The resulting transformation had the following parameters:

$$T_x(x,y) = 8.5256 * x - 2.5192 * y + 3429.44$$
$$T_y(x,y) = -1.4883 * x - 6.2442 * y + 13177.Ø8 \qquad (8.5)$$

Visual inspection of some projected objects verified that the accuracy was within the expected maximum allowance of about 11 pixels.

8.4.3 Results with procedure SHIFT

Procedure SHIFT was applied to the first 13 objects of the list (figure 8.36) sorted by decreasing object sizes. The correlation measure was calculated between the binary matrix representing the object and a compressed image which was preprocessed according to the feature type.

The maximum shift along the coordinate axis was chosen as 11. In accordance with the papers mentioned above to calculate the transformation parameters the error of such a transformation can amount to about 11 pixels or less.

Figure 8.37 lists the results. They are sorted by decreasing correlation values so that the best fitting of an object with the image stands on the top (FORST). The resulting relative shift vector to attain the maximum correlation from the start projection is used to calculate control points for the objects with a correlation greater than 50 %.

Feature	Correlation [%]	Optimum shift (col.,scan lines)
FORST	68.667	(- 5 , +1)
KAISERWALD-S	66.228	(-11 , -1)
WUNDSCHUH	64.6Ø3	(- 8 , +1)
KAISERWALD-N	6Ø.159	(- 7 , +2)
KOLLISCH	36.379	(- 7 , -1)
MURFELD2	34.459	(- 3 , +3)
SCHOTTERGRUB	33.834	(- 6 , +2)
MESSENDORF	3Ø.945	(-10 , +2)
EGGENBERG	25.285	(- 1 , +5)
GEIDORF	17.4Ø3	(- 9 , +3)
WALTENDORF	15.957	(-11 , +2)
ROSENBERG	15.327	(- 1 , +2)
MURFELD1	9.335	(+ 3 , +4)

Figure 8.37: Results of procedure SHIFT for GRAZ.

Only four objects fulfil this verification condition: FORST, KAISERWALD-S, WUNDSCHUH and KAISERWALD-N. These are all of type forest (5). A subsequent optical inspection of the image confirmed the following experience:

Objects can be automatically recognized on digital satellite images if the object can be clearly distinguished by a human interpreter.

This experiences is obvious of course and is reflected in publications on pattern recognition. It must be borne in mind in the present purpose.

An important reason for the small correlation values can be found in the map data bank. WALTENDORF for example is a political district, but was erroneously coded as a "built-up-area"; it could never be recognized in a satellite image. KOLLISCH and ROSENBERG have changed their shape because the map has not been revised to reflect recent urban growth.

We find thus that digitization of current topographic maps can lead to an inappropriate data bank for automatic feature recognition. It needs to be complemented by human intelligence to flag features that can predictably not be recognized.

For comparison reasons the four recognized objects are used to compute a new 6-parameter transformation by least squares. Figure 8.38 lists the resulting residuals and the corresponding standard deviations.

Feature	Error vector	Length	Correl.
KAISERWALD-N	(-Ø.362 , -Ø.711)	Ø.798	6Ø.159 %
KAISERWALD-S	(Ø.466 , Ø.915)	1.Ø27	66.228 %
WUNDSCHUH	(-Ø.329 , -Ø.645)	Ø.724	64.6Ø3 %
FORST	(Ø.225 , Ø.441)	Ø.495	68.667 %
Stand.Deviation	(Ø.712 , 1.398)	1.569	64.914 %

Figure 8.38: Residuals of the four objects recognized by SHIFT.

8.4.4 Results with procedure THRESH

Several parameters and configurations could be determined in project WALCH. From the experiment with lake WALCHEN, where we eliminated pixels of the pattern after the threshold operation according to their distance to the map projection, we learned to restrict the threshold operation to a starting mask. This starting mask has an equivalent effect of not taking into account pixels far away from the projected object. By this change the philosophy of the grey bound determination must be adapted.

The question arises what factor p should be used (line 6 of procedure THRESH, figure 6.3a). Since the resulting pattern should have at least as many pixels as the projection, the factor must be at least 1.Ø. For the upper bound we empirically chose 1.5. This interval [1.Ø, 1.5] had to be tested.

Figure 8.39 lists the resulting correlation values for p values from 1.Ø to 1.5 with step Ø.Ø5 and for eight objects. These objects have been chosen according to previous experiences (forests, optically distinguishable objects). Maximum correlation is achieved for different p values, but correlation values vary very little so that the objects recognized at the maximum are also recognized for every p in the chosen range. In the same way the corresponding control point differences remain less than 1 pixel. Therefore the value p = 1.1Ø was selected to be a good compromise for the four recognized objects (p = 1.1Ø minimizes correlation differences with least squares).

p-value / Feature	1.ØØ	1.Ø5	1.1Ø	1.15	1.2Ø	1.25	1.3Ø	1.35	1.4Ø	1.45	1.5Ø
KAISERWALD-S	78.729	79.Ø46	78.915	77.782	76.516	74.77Ø	73.264	71.32Ø	7Ø.449	69.31Ø	68.24Ø
WUNDSCHUH	72.728	7Ø.989	68.423	67.533	64.555	62.1Ø5	6Ø.816	59.321	58.913	56.556	55.867
FORST	63.689	64.279	65.874	65.732	66.392	66.19Ø	64.Ø72	63.322	64.385	62.6Ø9	61.793
KAISERWALD-N	54.897	55.531	57.155	58.268	58.668	59.48Ø	6Ø.Ø93	6Ø.893	61.553	61.654	61.478
PUCHWERK	44.447	43.722	47.624	47.285	48.418	48.567	49.988	49.35Ø	49.1Ø2	47.111	46.Ø13
SCHLOSSBERG	44.721	44.721	47.315	37.64Ø	37.64Ø	37.67Ø	36.647	39.973	39.973	38.21Ø	38.21Ø
KOLLISCH	36.ØØ3	36.554	37.6Ø8	37.235	37.95Ø	38.38Ø	38.569	39.597	39.67Ø	4Ø.Ø13	4Ø.5Ø3
GREITH	3Ø.Ø25	31.255	31.3Ø8	32.616	32.631	32.533	34.6Ø1	34.352	36.772	37.561	38.987

Figure 8.39: Testing parameter p of procedure THRESH by correlating (in %) with the map projection.

The centers of gravity of the four "recognized" objects are then used to calculate a 6-parameter transformation. Results are shown in figure 8.4Ø. No explanation could be found for the fact that the first component (=position within a scan line) of the error vector is greater than 1 pixel while the second component is always smaller than 1 pixel.

Feature	Error vector	Length	Correl.
KAISERWALD-N	(-1.949 , -Ø.Ø86)	1.951	57.155 %
KAISERWALD-S	(2.5Ø8 , Ø.111)	2.51Ø	78.915 %
WUNDSCHUH	(-1.768 , -Ø.Ø78)	1.77Ø	68.423 %
FORST	(1.2Ø9 , Ø.Ø54)	1.211	65.874 %
Stand.Deviation	(3.831 , Ø.17Ø)	3.835	67.592 %

Figure 8.4Ø: 6-parameter transformation with control points from procedure THRESH.

8.4.5 Experiences of tests with procedures LISU and LINDET

Two dominant linear features appear in the area of interest:

- river MUR and
- two intersecting highways (AUTOBAHN)

Figure 8.41 shows for example the "mapline" (see figure 6.6) of the river MUR and the corresponding starting mask.

Linefollowing worked for river MUR on the complementary image of figure 5.18. This makes sure that pixels on the river get high values. The following cost functions are results of a histogram over the starting masks:

$$cg(g) = (25.Ø8 - Ø.Ø9796 * g)^{2.21} \quad \text{for MUR}$$

and

$$cg(g) = (48.2977 - Ø.1886 * g)^{1.94} \quad \text{for AUTOBAHN} \qquad (8.6)$$

With these parameters procedure LISU was applied to the processed images. The search for the highways was split into 4 separate searches in a way that the end point was always the intersection. After the searches the 4 parts of AUTOBAHN ("binline") have been joined together for illustration purposes (figure 8.42). The development area is shown in figure 8.43.

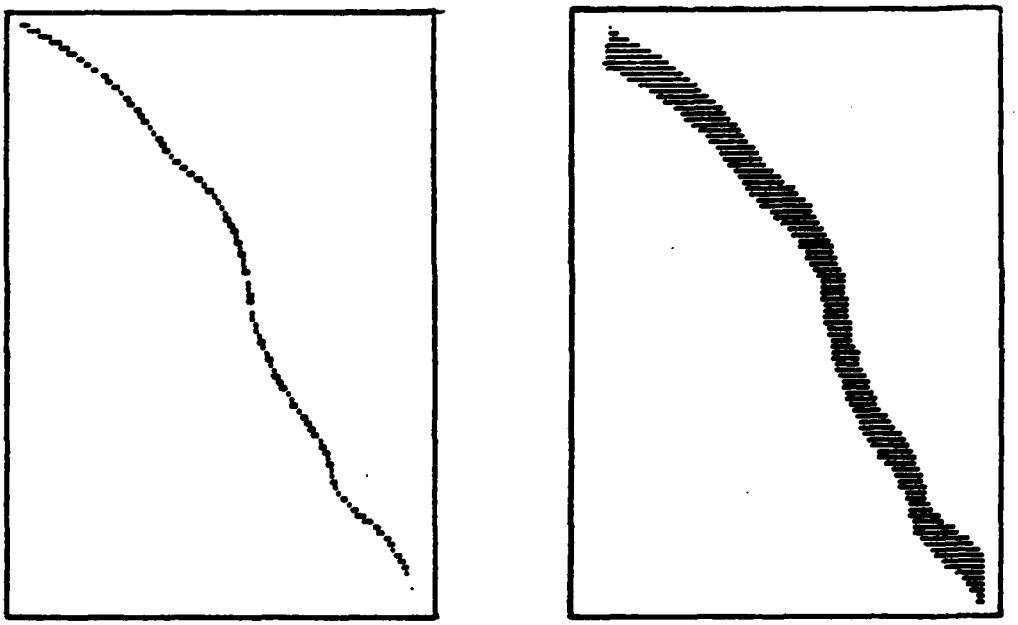
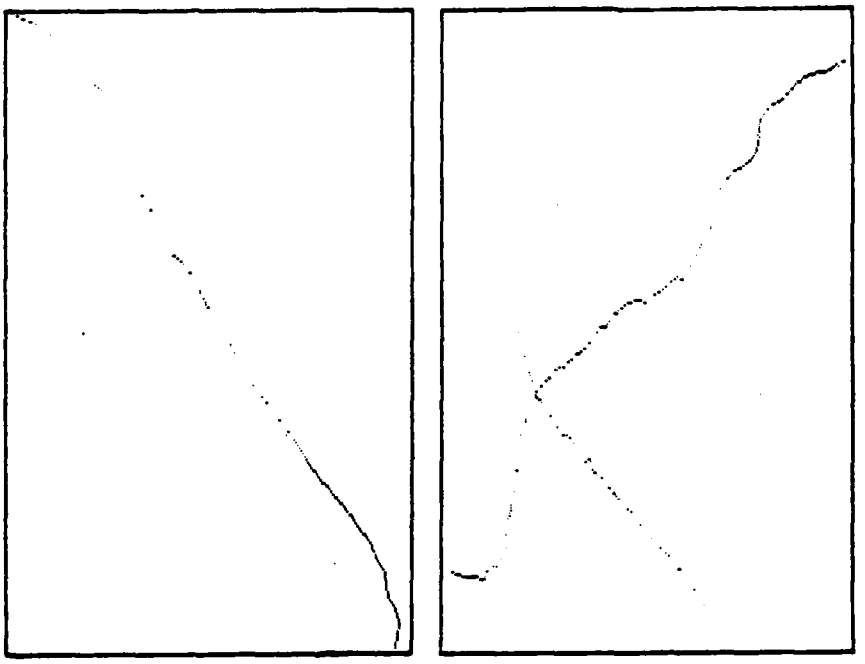


Figure 8.41: (a) Mapline MUR and (b) corresponding starting mask.

The performance of procedure LISU for the two feature types can be seen in Table 8.5 :

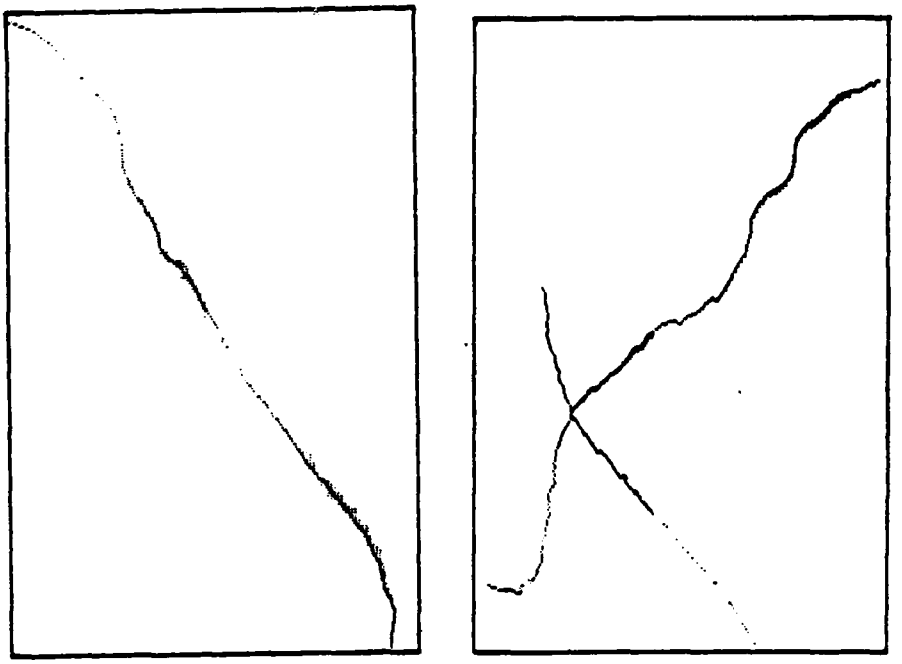
Feature	Length of binline	Surface of development area	Computing time
MUR	282 pixel	1994	278.9 sec
AUTOBAHN	48Ø pixel	1818	71.9 sec

Table 8.5 : Performance of procedure LISU with two different features.



(a) (b)

Figure 8.42: Resulting binary matrices of (a) river MUR and (b) highways AUTOBAHN.

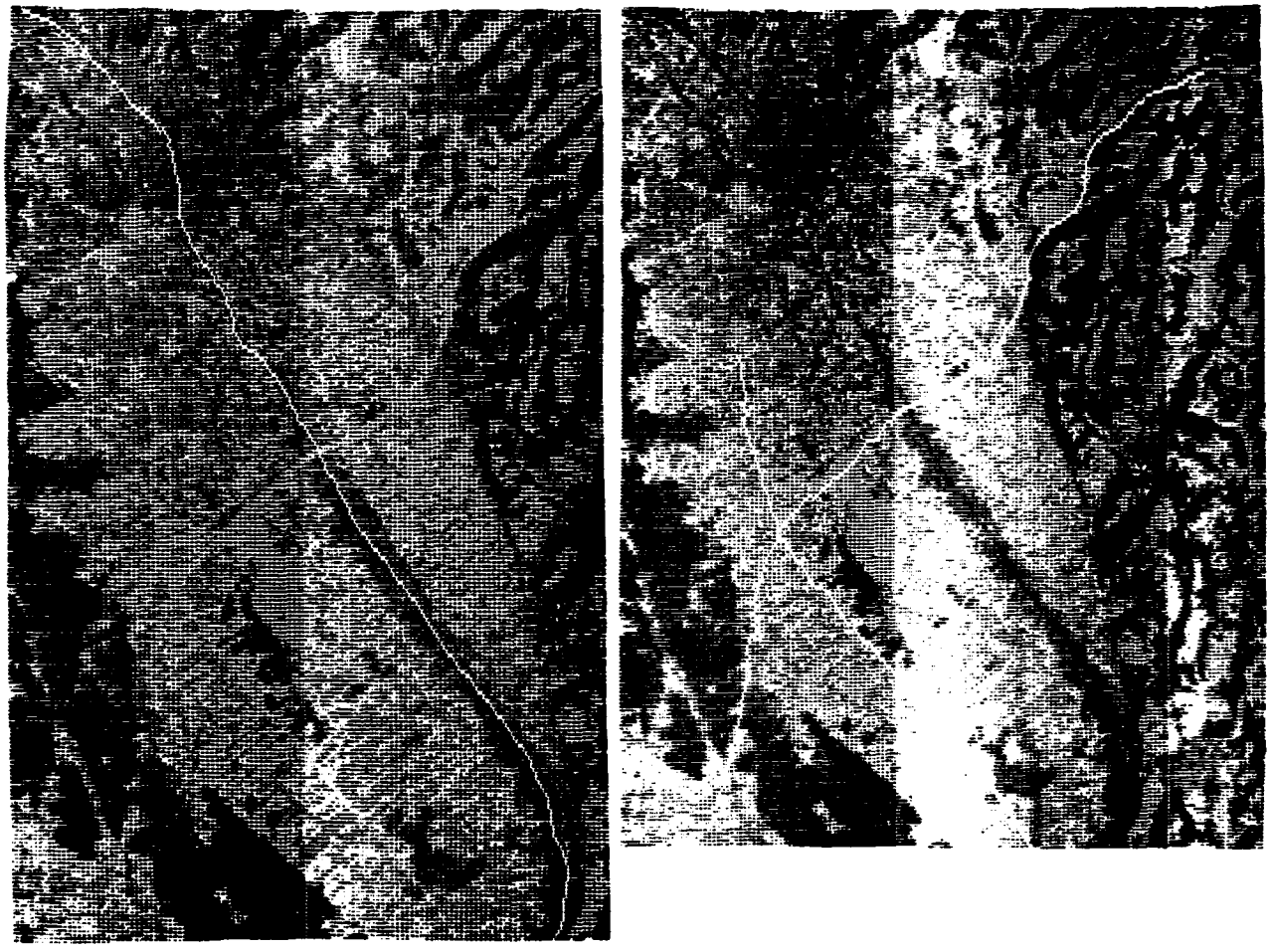


(a) (b)

Figure 8.43: Development areas of (a) river MUR and (b) highways AUTOBAHN.

The computing time for AUTOBAHN is the time-sum for the four segments. Computing time in LISU is not proportional to the length of the line, it grows with a high order dependency because set S of continuation candidate pixels is growing with increasing line length and must be inspected completely for every step towards the goal.

For optical verification the resulting binary matrices have been printed out together with the compressed image of GRAZ (figure 8.44).



(a) (b)

Figure 8.44: Optical verification of procedure LINDET, for MUR (a) and for AUTOBAHN (b).

Computational verification by subdividing both "mapline" and "binline" into equal length segments for which a 6-parameter transformation is computed shows that the position of the starting and end point has great importance for automatic verification measures. Figure 8.45 lists the result (error vectors and lengths) for a segment of AUTOBAHN and for MUR with 5 segments each. Starting and end points have been eliminated according to experiences from the synthetic image.

Feature	Error vector	Length
MUR	(1.879 , -Ø.271)	1.898
	(Ø.534 , -Ø.883)	1.Ø32
	(Ø.39Ø , Ø.Ø95)	Ø.4Ø1
	(-2.2Ø5 , Ø.852)	2.364
AUTOBAHN	(-Ø.516 , -Ø.Ø46)	Ø.519
	(-2.395 , 1.Ø62)	2.62Ø
	(Ø.472 , Ø.Ø71)	Ø.477
	(1.842 , -Ø.879)	2.Ø41
Stand.Deviation	(1.921 , Ø.836)	2.Ø95

Figure 8.45: Verification of linear features by segmenting.

8.4.6 Some results with procedure ADAPT

With procedure SHIFT and THRESH only 4 objects could be recognized supporting a starting deviation of the first projection of about 11 pixels. This estimated deviation from the real position of the feature becomes better if the results of procedures SHIFT and THRESH are used to calculate the transformation. Control points of LINDET have not been used because it could not be measured how good the match of corresponding points has been.

The resulting transformation has the following form:

$$T_x(x,y) = 8.72 * x - 2.Ø8 * y + 2492.32$$
$$T_y(x,y) = -1.57 * x - 6.26 * y + 13227.4 \qquad (8.7)$$

From the residuals of the least squares solution discrepancies of about 4 to 6 pixels have been found.

Another factor that leads to increases of computing time of ADAPT was the surface of the feature. So 5 objects smaller than for procedures SHIFT and THRESH were used: WUNDSCHUH, GREITH, FLIEGERHORST, PUCHWERK and THALERHOF.

The version of ADAPT as described in chapter 6.4 has been applied to these 5 objects. Figure 8.46 lists the pixel changes (added +, eliminated -) for each iteration:

	WUNDSCHUH		GREITH		FLIEGERHORST		PUCHWERK		THALERHOF	
Iteration	+	-	+	-	+	-	+	-	+	-
1	57	21	33	2Ø	11	6	39	Ø	15	Ø
2	35	27	19	23	1Ø	2	4Ø	Ø	14	Ø
3	23	29	11	27	12	1	42	13	15	Ø
4	1Ø	32	2	31	15	Ø	43	7	17	Ø
5	4	44	-	-	-	-	46	7	-	-
6	3	38	-	-	-	-	47	12	-	-
Correl.[%]	72.Ø31		71.183		62.655		49.337		46.852	

Figure 8.46: Adapting 5 objects to the satellite image.

The last line shows that the first 3 objects have been recognized successfully. THALERHOF was too small to allow a correct grey value determination by the automatic routine. The range was therefore toolarge and pixels had not been eliminated during ADAPT. A similar effect accured for PUCHWERK. A reason why FLIEGERHORST could be verified although it is smaller than PUCHWERK was found in the bordering forest. It gives a good contrast to the feature so that the automatic grey value determination delivered a correct interval.

With three control points from ADAPT, no overdetermination was given for a 6-parameter transformation for ADAPT and therefore no residuals and standard deviation could be calculated.

8.4.7 Rectification of image GRAZ

With the aid of the recognised control points procedure RECTIFY was applied to the four channels of the original satellite image. Discrepancies between control points of method SHIFT and method THRESH have been eliminated by using the result with the better verification measure. Figure 8.47 can be regarded as the final result of the automatic registration process on GRAZ.

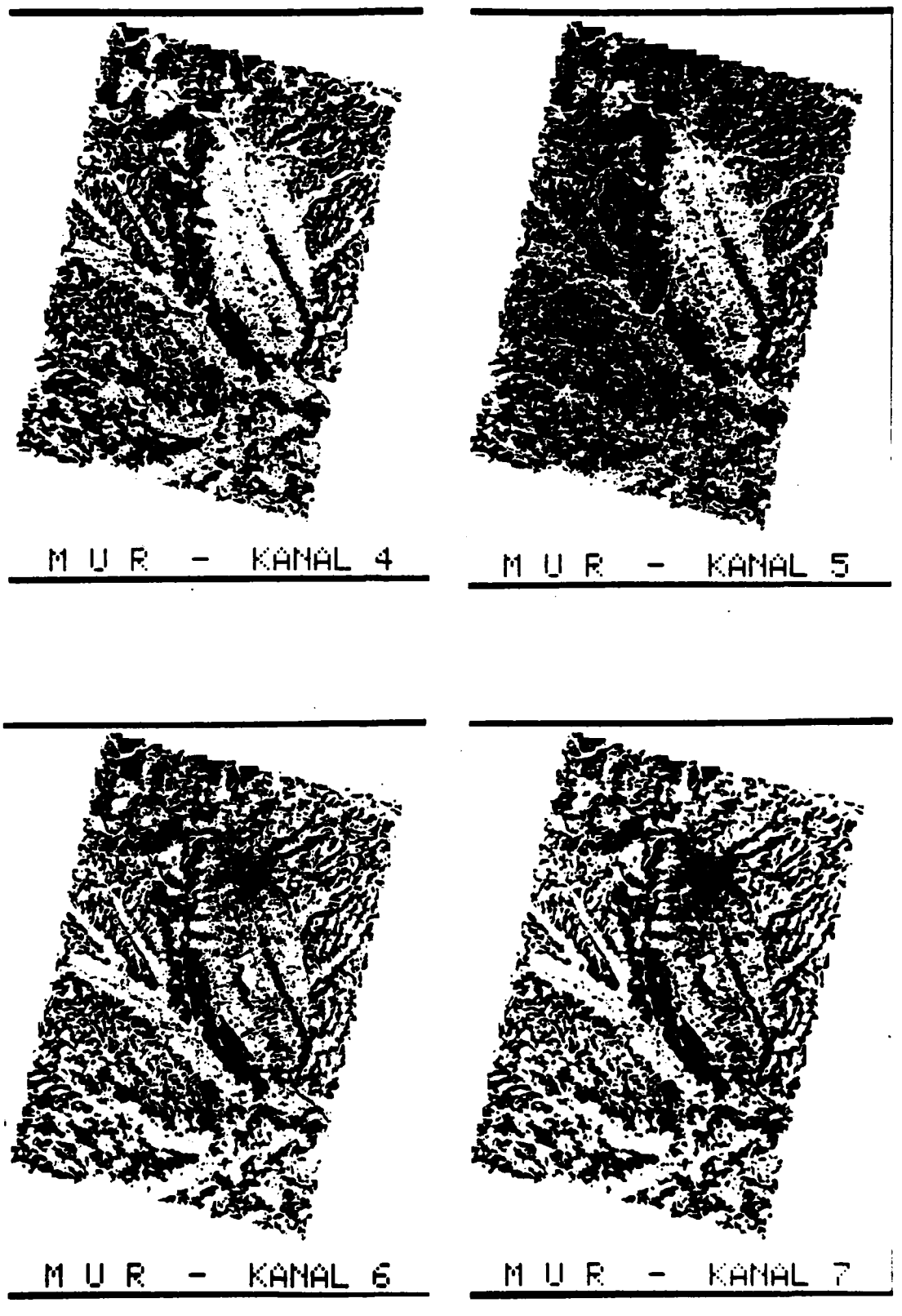


Figure 8.47: Rectified satellite image of GRAZ.

9. CONCLUSIONS AND OUTLOOK

This report describes tests which have been performed to verify a general concept for automatic registration of a digital map with a satellite image. Programs and methods have been developed for this purpose. The versions of algorithms as described here are products of a development process and are not operational. They only demonstrate the principle of our concept and the fact that it works under certain constraints.

Future research should be concentrated on optimization of the general performance of each of the proposed procedures and the development of an appropriate processing system. New procedures can then be included in the modular concept.

From the experiences, particularly with image GRAZ, the following suggestions can be derived regarding the required processing system.

A processing system for automated feature recognition:

A processing system is required that implements procedure REGISTER in more than one step. Typically an initial step should be based on satellite orbit data and large, distinct map objects, using a simple and fast recognition method. This result in a few control points which can be used, together with flight data, to compute a better approximation for the projection function. One can then more efficiently employ more complex methods and use smaller and therefore also more features.

The processing system must implement a carefully designed procedure OBJECTSELECTION that incorporates probabilities of being able to recognize a feature. Political districts are of course inappropriate for the purpose. A learning capability should be included. Sorting criteria should be type of feature, size, contrast with surroundings, predictability of its appearance in the image, scores from past uses.

Pre- and postprocessing functions:

Preprocessing must be more adapted to the used recognition procedures and object types. Texture classification should be added. Tests should be done to study the effect of some scale reducing preprocessing (parsing). Thresholding depends very much on preprocessing. Therefore typical features should be associated with characteristic preprocessing. We did test some postprocessing after thresholding (image WALCH, object WALCHENSEE) by a distance transform of the intersection of the projection and threshold masks to eliminate obvious noise.

Post-processing after initial pixel-selection is thus of particular interest in THRESH.

Reprogramming of more operational versions of recognition techniques:

It is in the nature of research and development that computer programs are general, flexible and slow. Numerous ideas exist and could now be implemented to considerably improve the throughput times for all investigated algorithms. The improvements can be in the software, using appropriate computing strategies (compare brute-force similarity detection via correlation factors and a sequential similarity detection as proposed by BARNEA and SILVERMAN, 1972). They can be in hardware, using e.g. pipeline and/or array processing equipment.

To give just one example, one can consider line-following software where there is considerable room for improvements, limiting set S. Even a small change such as eliminating points from S based on time when they were accepted in S can be of great value. This would lead to an algorithm similar to that of MONTANARI (see VANDERBRUG, 1973) where backtracking is limited to the last three levels.

Verification:

Verification is very rudimentary at the moment. Rotation and scale should be incorporated in the verification process to yield better discrimination measures for similarity of binary matrices. One possibility can be to shift the centers of gravity to be on the same place and then to correlate the two shapes.

If verification improves, then more than a single control point can be extracted from a single object.

In spite of the fact that software is thus far from optimized we can conclude from the experiences that ARSIM is a valid concept for the computer-supported registration of maps and images. Limitations of the success are caused by the limitation of experiences and the map data bank. We feel that the proposed approach to the problem is flexible, modular and broad so that registration can be done in the optimum way. The problem is then to define this optimum. This, however, is a question of experimentation and learning.

Recently it has become evident, that the combined use of images and maps is a necessity to study complex environmental and other questions. The ARSIM-concept may thus be of interest to a broad range of applications, even where control-point definition is at first sight not a problem of automation. Ultimately it will be of great interest to guide the image analysis using map information. The inherent differences between maps and images can then be handled through the logic that is implemented in ARSIM.

10. REFERENCES

ALEKSANDER I. (1978): Improving Patterns of Recognition, Computing Europe, 6.Febr.1978, p.18.

ANUTA P. (1970): Spatial Registration of Multispectral and Multitemporal Images Using Fast Fourier Transform Techniques, IEEE Trans. on Geoscience Electronics, Vol. GE-8, No. 4.

ANUTA P. (1973): Geometric Correction of ERTS-1 Digital Multispectral Scanner Data, LARS Information Note 103073, Purdue Univ., West Lafayette, Indiana, USA.

BARNEA D., I. SILVERMAN (1972): A Class of Algorithms for Fast Digital Image Registration , IEEE-Trans. on Computers, Vol. C-21, No. 2.

BARROW H.G. et al. (1977): Experiments in Map-Guided Photo-Interpretation, Proc. 5th Intl. Joint Conf. on Artif. Intell.-IJCAI-77, Mass.Inst. of Techn., Cambridge, Mass., August 22-27, 1977.

BRÜGGEMANN H. (1978): Flächenbezogene graphische Datenverarbeitung-Programmentwicklung beim Landesvermessungsamt Nordrhein-Westfalen, Nachrichten aus der Karten- und Vermessungswesen, Reihe I, Heft Nr. 75, S. 33-49, Frankfurt am Main.

BRYANT N., A. ZOBRIST (1977): IBIS: A Geographical Information System based on Digital Image Processing and Image Raster Data Type, IEEE, Trans. Geoscience Electronics, Vol. GE-15, No. 3.

COLVOCORESSES A.P., R. McEWEN (1973): EROS Cartographic Progress, Photogrammetric Engineering, Vol. 39, No.12.

DAVIS L.S. (1973): A Survey of Edge Detection Techniques, Computer Science Center, TR-273, Univ. of Maryland.

DOWMAN I., A. HAGGAG (1977): Digital Image Correlation Along Epipolar Lines, Proc. Intl. Symp., Image Proc., Graz, Austria.

DYER C.R., A. ROSENFELD, H. SAMET (1979): Region Representation: Boundary Codes from Quadtrees, NATO ASI Lecture Notes-June, Maratea, Italy.

FLONZAT G. et al. (1979): Extraction de l'Information d'une image LANDSAT contenue dans un contour artificiel, 3rd GDTA Symposium, Toulouse, 19 - 22 June 1979.

FRANK A. (1979): "Datenbanken", Vermessung, Photogrammetrie, Kulturtechnik, 77. Jahrgang, Heft 7-8, S.205-210, Zürich.

FREEMAN H. (1979): Map Data Encoding Techniques, NATO ASI Lecture Notes-June, Maratea, Italy.

GOODCHILD M.F. (1979): Geographical Data Capture and Encoding, NATO ASI Lecture Notes-June, Maratea, Italy.

HARRIS J.F. (1979): Automatic Map Digitizing - A Taxonomy of Available Technology, NATO ASI Lecture Notes-June, Maratea, Italy.

HARRIS J.F., G. PRESTON (1979): Automatic Map Digitizing-Systems and Algorithms, NATO ASI Lecture Notes-June, Maratea, Italy.

HOLDERMANN F.(1971): Processing of Grey Scale Pictures, Forschungsinstitut f. Informationsverarbeitung and Mustererkennung (FIM), Karlsruhe, BRD.

KELLY M.D. (1971): Edge Detection in Pictures Using Planning, Machine Intelligence 6, Edinburgh, Univ.Press, Scotland.

KONECNY G., W. SCHUHR (1975): Digitale Entzerrung der Daten von Zeilen-Abtastern, Bildmessung und Luftbildwesen, 4, 1975.

KREILING W.(1976): Automatische Auswertung in Stereobildern durch digitale Korrelation, Pres.Paper, Comm. III, 13th Congress of ISP, Helsinki, Finland.

KREYSZIG E. (1973): Statistische Methoden und ihre Anwendungen, Vandenhoeck & Ruprecht, Göttingen, BRD.

KROPATSCH W., F. LEBERL (1978a): Concept for the Automatic Registration of Satellite Images with a Digital Map Data Base, Proc. Symp. of Comm. IV, Intl. Soc. Photogrammetry, Ottawa, Canada, 6-8 Oct.1978, pp.411-424.

KROPATSCH W., F. LEBERL (1978b): Edge and Line Detection in Digital Images, Symp. of Comm. III, Intl. Soc. of Photogrammetry, Moscow, USSR, 19 July - 5 August 1978.

LEBERL F., W. KROPATSCH (1979): Map-Guided Automatic Analysis of Digital Satellite Images, Mittlg.d. Geod. Institute d. Techn.Univ. Graz, Folge Nr. 33.

MALILA W.A., R.H. HIEBER, A.P. McCLEER (1973): Correlation of ERTS MSS Data and Earth Coordinate Systems, Environmental Research Institute of Michigan Ann Arbor, Michigan, USA.

MONTOTO (1977): Digital Detection of Linear Features in Satellite Imagery, Proc. Intl. Symp. Image Processing, Graz, Austria.

NACK M.L. (1975): Temporal Registration of Multispectral Digital Satellite Images using their Edge Images, AAS/AIAA Astrodynamics Specialist Conference, Nassau, Bahamas, Paper No. AAS-75-104.

NEWMAN W.M., R.F. SPROULL (1979): Principles of Interactiv Computer Graphics, McGraw Hill Book Co, Series in Artificial Intelligence, Tokyo, 541 p.

NILSSON N.J. (1971): Problem Solving Methods in Artifical Intelligence, Mc Grac Hill Book Co., Brüssel.

PRATT W.K. (1978): Digital Image Processing, Image Enhancement, Publ. by Wiley & Son, 1978.

ROOS M. (1975): Automatic Reseau Measuring Equipment (ARME), Photogr. Eng. & Remote Sensing, Vol. 41, pp. 1109-1115.

ROSENFELD A.(1970): Connectivity in Digital Pictures, Journal of the Assoc.for Comp. Machine, Vol. 17, No. 1, pp. 146-160.

ROSENFELD A., L. DAVIS (1975): A Note on Thinning, TR-381, Comp. Science Center, Univ. of Maryland, USA.

SAKAROVITCH M. (1975): Introduction à l'étude des Graphes, Université Scientifique et Médicale de Grenoble - E.N.S.I.M.A.G.

STIES M., B. SANYAL, K. LEIST (1977): Organization of Object Data for an Image Information System", Research Institute for Information Processing and Pattern Recognition (FIM), Karlsruhe, FRG.

SWEDLOW M., C. McGILLEM, P. ANUTA (1978): Image Registration: Measure and Preprocessing Method Comparisons, IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-14, No. 1.

TENENBAUM J.M., M. TISCHLER, H. WOLF (1978): A Scene Analysis Approach to Remote Sensing", Technical Note 173, Artificial Intelligence Center, Stanford Research Institute, Menlo Park, California, USA.

VANDERBRUG G.J. (1973): Edge Following in Noisy Pictures, Computer Science Center, TR-270, Univ. of Maryland.

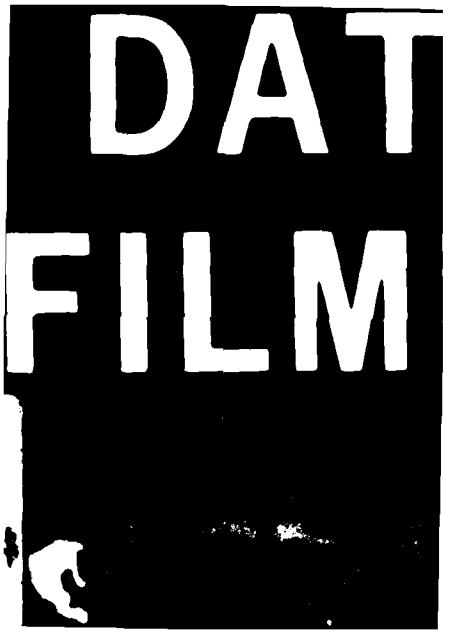
VAUQUOIS B. (1970): Calculabilité des Langages, Université de Grenoble, France.

VEILLON G. (1974): Algorhtmique, Univ. Scientifique et Médicale de Grenoble, 2éme année E.N.S.I.M.A.G.

WEBER W.(1978): Geographische Informationssysteme - ein Überblick und Gedanken zur weiteren Entwicklung, Nachrichten aus dem Karten- und Vermessungswesen, Reihe I, Heft Nr. 75, S.159-186, Frankfurt am Main.

WESZKA J.S., C.R. DYER, A. ROSENFELD (1979): A Comparative Study of Texture Measures for Terrain Classification; NATO ASI Lecture Notes-June, Maratea, Italy.

WIESEL J. (1977): DIDAK, A Digital Image Processing System, Proc. Intl. Symp. Image Processing, Graz, Austria.


DAT
FILM